intel

APPLICATION NOTE

AP-262

October 1986

Using Procedures to Speed I²ICE™ System Debugging

LAKSHMI JAYANTHI
DSO APPLICATIONS

Order Number: 280722-001

INTRODUCTION

Every engineering manager in charge of developing a microprocessor based product worries about the critical phase of the project when the hardware and software pieces are integrated. The integration portion of a development task can take as much as 40 percent of the design team's time, and some projects have been known to die at this point, succumbing to problems too expensive to solve.

Today's high-peformance 16-bit microprocessors make the integration phase more critical than ever. The move from 8-to 16-bit microprocessors has added considerable complexity to bus structures and memory management techniques. These processors address such large amounts of memory, the 80286 can access 16 megabytes of physical memory, that development of a product based on such a processor is software-intensive.

This additional complexity makes crucial the need for tools that can help hardware and software integrators gather, correlate, and evaluate data. Integrating the hardware and software of the microprocessor based system often ends in failure because of a classic communication gap between hardware and software designers. Each uses different methods to debug the system. The gap has been narrowed by such software development aids as an In-Circuit emulator (ICE™) system, PSCOPE, and TargetSCOPE-186 and such hardware tools as logic analyzers, but a full solution calls for a common interface that enables designers to go back and forth between hardware and software domains.

Such an interface is the I²ICE™ (Integrated Instrumentation And In-Circuit Emulator) System. This new system integrates a logic analyzer, a high-level software debugger, and a sophisticated in-circuit emulator, permitting hardware and software developers to interact in the debugging process without learning each others language.

INTEGRATED INSTRUMENTATION AND IN-CIRCUIT EMULATOR (I²ICE system) is a very powerful Hardware and Software debugging tool used to debug the hardware board or the software code. In order to achieve the debugging you need, numerous I²ICE system features are available such as the following:

- Integrated Command Directory (ICD)
- Coprocessor Support
- Pseudo-variables
- Debug Procedures
- Logic Clips
- Logic Timing Analyzer

This application note focuses on the various ways to write debug procedures with an I²ICE system for hardware and software debugging purposes. In general this note deals with procedures that can be used on an I²ICE system. The numerous techniques that can be used in creating and developing these procedures are explained in detail with specific examples, flow charts, diagrams, displays, and actual procedure listings.

USEFUL DEFINITIONS

Before looking into the actual procedural language, it is extremely helpful to study some useful definitions. It is important to understand these definitions and their usage in the I²ICE system in order to understand their role in the development of procedures. Each relevant command is defined and explained in some detail with examples. At the end of this section you will understand these definitions and their role in the procedural language.

Procedure

A procedure operates like a single command because it enables you to use several commands in a block structure and declare local variables. Also, the procedures can be several nested blocks. The size of procedures is limited only by the amount of memory space available. Defining procedures is like adding commands to the I²ICE language and you can create commands of particular relevance to your debug situation.

INCLUDE

The INCLUDE command retrieves a command file from a mass storage device and loads it into memory.

LIST

A LIST file is an FICE system utility file. Typically, a list file is used as a debug session log.

LITERALLY

LITERALLY definitions are abbreviations for previously defined character strings.

WRITE

The WRITE command is most often used in procedures to add explanatory text to returned values in a more useful form, such as a table.

Attribute Codes

Attribute codes are used to add clarity or distinction to text written to the screen. These attribute codes are accessible using the CONCAT command. The CONCAT command builds strings by concatenating all or parts of old strings to form a new string.

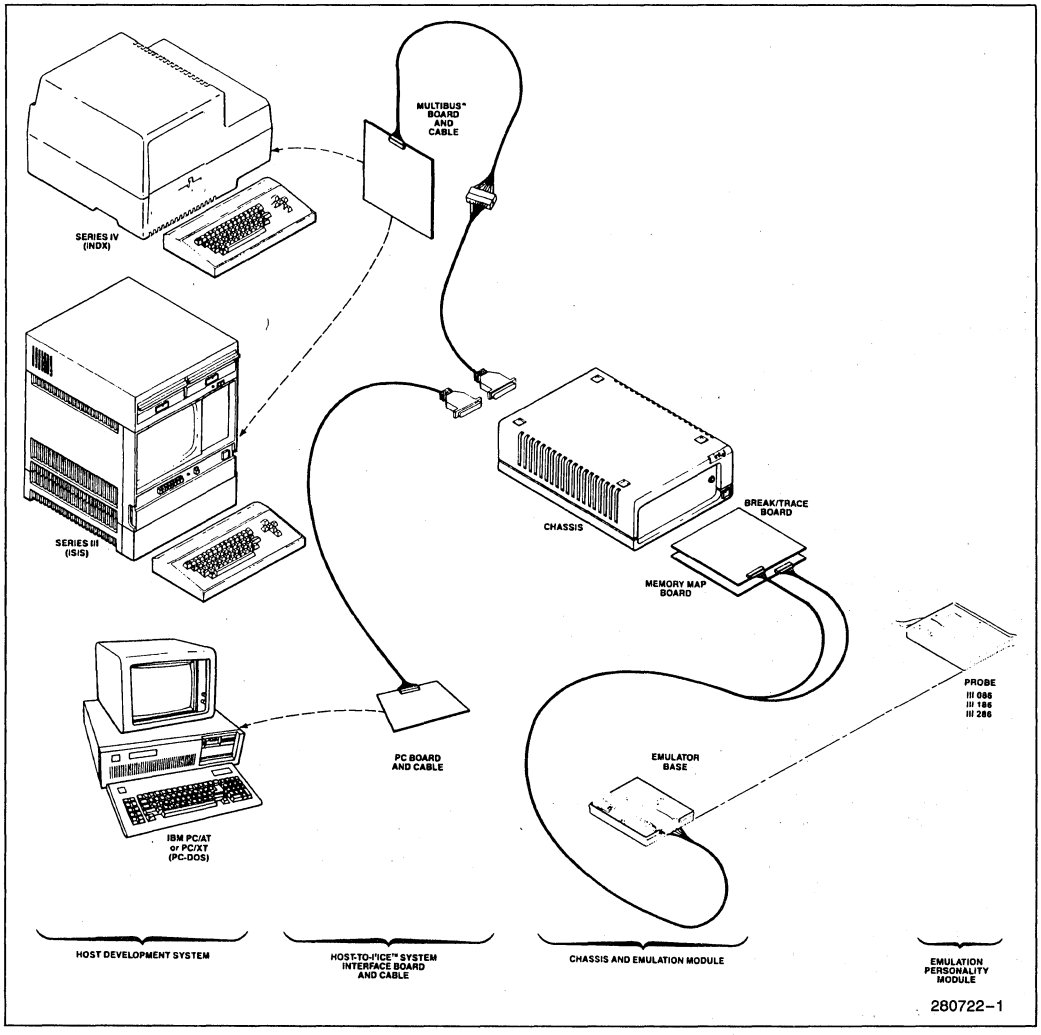


Figure 1. I²ICE System

Boolean-Condition

A Boolean condition is either a value of type BOOLEAN (TRUE or FALSE) or an expression that uses one of the relational operators:

CI

The CI (console input) function enables a debug procedure to read one character from the system terminal.

IF

An IF command conditionally executes a command or group of commands.

PROBLEM DEFINITION

Now that we understand some of the relevant I²ICE definitions, we may move to the problem definition. All the preceding definitions will be used with examples of

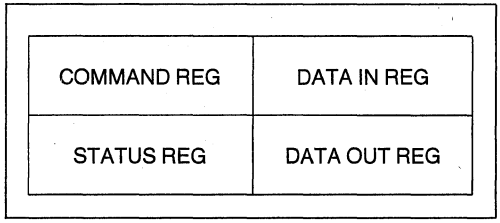


Figure 2. Registers in Asynchronous Mode

the procedural language. Debug procedures will be built-up in a variety of forms and their tremendous leverage and flexibility will be demonstrated. The countless number of ways commands like INCLUDE, LIST, WRITE are used and the numerous functions that they perform will be illustrated.

In this example we will debug a program which contains driver code interacting with peripheral components. The interaction with two of these peripherals (the simple 8251A Programmable Communications Interface and the more complex 82530 Serial Communications Controller) will be dealt with and also how these debug procedures can simplify and speed our work will be shown. We will develop techniques for working with these components and reduce frustration and constant thumbing through data catalogs.

The 8251A Programmable Communications Interface is a simple component as viewed by the software designer. It consists (in Asynchronous mode) of four registers as shown in Figure 2. When we read the status port, the multiple bits must be decoded to know what the 8251A is doing. Our first debug procedure will do this decoding for us and we will never confuse the bits again.

8251A Procedure

Prior to starting data transmission or reception, the 8251A must be loaded with a set of control words generated by the CPU. These control signals define the complete functional definition of the 8251A and must immediately follow a reset operation (internal or external). The control words are split into two formats: mode instruction and command instruction.

In data communications it is often necessary to examine the "status" of the active device to ascertain if errors have occurred or other conditions that require the processor's attention. With the 8251A facilities, the programmer can "read" the status of the device at any time during the functional operation.

The following procedure executes a status read of the 8251A and displays it on the I²ICE system screen. The status read of the 8251A will be at some port location depending on the hardware configuration of the 8251A. This procedure, by way of example, requests the user to enter the port address of the 8251A from which the status information can be read. Operationally, you would define this once as a GLOBAL variable. The procedure then displays the byte value stored at that memory location in binary form on the screen. The corresponding messages for each of those eight bits with complete information about their status appears on the screen.

There are two procedures under one filename. They are GETHEX and I8251A, respectively. GETHEX is a procedure used to change the user input at the console to a hexadecimal value, to print this value on the screen, and to ring a bell if you make an improper selection. I8251A is the main procedure that reads in the hexadecimal value of the port address of the 8251A and (after you choose the port number) displays the eight bits of information in binary form with detailed explanation. The actual display on the I²ICE System terminal screen is as shown in Figure 3.

```
*define PROC gethex = DO
*.define WORD num
*.define CHAR chr
*.define CHAR bell = 07h
*.num = 0
*.repeat
*..chr = ci
*..if (chr > = '0') and (chr < = '9') then
*...num = num*10h + (chr-30h)
*...write using ('1,>') chr
*...else
*....if (chr > = 'A') and (chr < = 'F') then
*....num = num*10h + (chr-37h)
*....write using ('1,>') chr
*....else
*.....if (chr > = 'a') and (chr < = 'f') then
*.....num = num*10h + (chr-57h)
*.....write using ('1,>') chr
*.....else
*......if chr < > 0dh then write using('0,>') bell
*......end
*.....end
*....end
*...end
*...until chr = = 0dh
*..end
*..return num
*.end
*define PROC I8251A = do
*.define WORD wreg
*.define WORD numget
*.define BYTE temp
*.curhome;cleareos
*.write concat(hibl,'ENTER THE PORT NUMBER OF THE 8251A and <cr>',norm)
*.wreg = gethex
*.temp = port(wreg)
*.curhome;cleareos
*.write concat(hi,'              8251A STATUS REGISTER',norm)
*.write concat(hi,'              **********************',norm)
*.write using ('2c," = >",2,y,5x,1,y,5x,1,y,5x,1,y,5x,1,y,5x,1,y,5x,1,y') &
**(temp and 80h) /80h, &
**(temp and 40h) /40h, &
**(temp and 20h) /20h, &
**(temp and 10h) /10h, &
**(temp and 08h) /08h, &
**(temp and 04h) /04h, &
**(temp and 02h) /02h, &
**(temp and 01h) /01h
*.write concat(norm,'  ---------------------------------------------------')
*.write concat(norm,' | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | ')
*.write concat(norm,'  ---------------------------------------------------')
*.write '   D7 = DATA SET READY   D3 = PARITY ERROR'
*.write '       D6 = SYNDET/BRKDET   D2 = TxEMPTY'
*.write '           D5 = FRAMING ERROR   D1 = RxRDY'
*.write'               D4 = OVERRUN ERROR   D0 = TxRDY'
```

280722-2

```
*.if (temp and 80h)/80h = = 1 then
*..write'DATA SET READY is SET'
*..end
*.if (temp and 80h)/80h = = 0 then
*..write'DATA SET READY is NOTSET'
*..end
*.if (temp and 40h)/40h = = 0 then
*..write'BRKDET is NOTSET — We have not received a break'
*..end
*.if (temp and 40h)/40h = = 1 then
*..write'BRKDET is SET — We have received a break'
*..end
*.if (temp and 20h)/20h = = 1 then
*..write'FRAMING ERROR is SET — A missing stop bit was detected'
*..end
*.if (temp and 20h)/20h = = 0 then
*..write'FRAMING ERROR is NOTSET — A missing stop bit was not detected'
*..end
*..if (temp and 10h)/10h = = 1 then
*..write'OVERRUN ERROR is SET — We have overwritten a previous character'
*..end
*.if (temp and 10h)/10h = = 0 then
*..write'OVERRUN ERROR is NOTSET — We have not overwritten a previous character'
*..end
*.if (temp and 8h)/8h = = 0 then
*..write'PARITY ERROR is NOTSET — We have not detected a parity error'
*..end
*.if (temp and 8h)/8h = = 1 then
*..write'PARITY ERROR is SET — We have detected a parity error'
*..end
*.if (temp and 4h)/4h = = 0 then
*..write'TxEMPTY is NOTSET — The characters have not been sent down the serial line'
*..end
*.if (temp and 4h)/4h = = 1 then
*..write'TxEMPTY is SET — All the characters have been sent down the serial line'
*..end
*.if (temp and 2h)/2h = = 0 then
*..write'RxRDY is NOTSET — A character is not ready to be input to the CPU'
*..end
*.if (temp and 2h)/2h = = 1 then
*..write'RxRDY is SET — A character is ready to be input to the CPU'
*..end
*.if (temp and 1h)/1h = = 0 then
*..write'TxRDY is NOTSET — The transmitter is not ready to accept a data character'
*..end
*.if (temp and 1h)/1h = = 1 then
*..write'TxRDY is SET — The transmitter is ready to accept a data character'
*..end
*.write concat(hibl,'              PLEASE HIT ANY KEY TO RETURN',norm)
*.temp = ci
*.end
```

280722-3

```
TYPE IN THE HEXADECIMAL STATUS READ ADDRESS FOR THE 8251A and <cr>

0H

0H
UNIT 0 PORT 0000H OUPUT BYTE 006H
UNIT 0 PORT 0000H REQUESTS BYTE INPUT (ENTER VALUE): 12

                8251A STATUS REGISTER
                **********************

=> 0    0    0    1    0    0    1    0
-----------------------------------------
| D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
-----------------------------------------
  D7=DATA SET READY    D3=PARITY ERROR
       D6=SNDET/BRKDET    D2=TxEMPTY
            D5=FRAMING ERROR    D1=RxRDY
                 D4=OVERRUN ERROR    D0=TxRDY

DATA SET READY is NOTSET
BRKDET is NOTSET - We have not received a break
FRAMING ERROR is NOTSET - A missing stop bit was not detected
OVERRUN ERROR is SET - We have overwritten a previous character
PARITY ERROR is NOTSET - We have not detected a parity error
TxEMPTY is NOTSET - The characters have not been sent down the serial line
RxRDY is SET - A character is ready to be input to the CPU
TxRDY is NOTSET - The transmitter is not ready to accept a data character
        PLEASE HIT ANY KEY TO RETURN
```

280722-4

Figure 3. 8251A Status Register Display

The 8251A is a very simple component. How could these techniques be used on a more complex chip such as the 82530 SCC? A procedures disk for the 82530 Serial Communications Controller (SCC) had been created (See Appendix B for availability). For those designs that do not utilize an 82530 SCC, it would still be beneficial to follow the text and deduce how the techniques developed could be used in your design. The purpose of creating this procedures disk is to minimize the usage of the 82530 Handbook. If you are using an 82530 chip in your design and also using an I²ICE system to debug your design board, then all you need to do is invoke this set of procedures onto the I²ICE system screen and all the required information will be given about the 82530. This drastically reduces the usage of the 82530 Handbook so that you can spend more time debugging and less hunting information.

The different techniques that have been used in creating this disk are explained in detail in the following sections. Refer to Figure 4 which shows a flow diagram which describes the interaction of the four main blocks representing four different procedures. They are MAIN.INC, SUPJOB.INC, JOB.INC and *.INC where the '*' indicates any of the subset procedures. Every procedure built has two segments of that corresponding procedure. An overlay, as well as an include version, has been created.

The overlay (OV0) version contains the main contents of the material while the include (INC) file contains information about the procedure invocation, execution, and removal. The include file will be explained in a later section.

PROCEDURES DESCRIPTION

Procedures for the following pieces of information have been created from the 82530 Handbook.

Apart from the preceding set of procedures, some main procedures have also been created which are instrumental in managing the entire set of procedures. Those procedures are as follows:

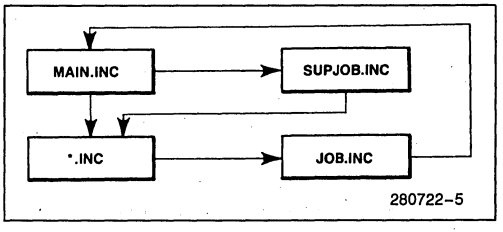


Figure 4. Flow Diagram

Handbook Description	Procedures Created	
• Pin Description	PINDES.OV0	PINDES.INC
• Register Function Description	REGDES.OV0	REGDES.INC
• Time Constant Values Description	TIMVAL.OV0	TIMVAL.INC
• Data Encoding Methods Description	DATENC.OV0	DATENC.INC
• Register Addressing Description	REGADD.OV0	REGADD.INC
• SCC Protocols Description	PRTCOL.OV0	PRTCOL.INC
• Register Bit Functions	PINFUN.OV0	PINFUN.INC

I²ICE.MAC—Initializes the drives and includes the main procedure, MAIN.OV0, and executes it.

MAIN.OV0—Handles the remainder of the procedural management. It contains the main core and acts as a supervisor.

MAIN.INC—Executes the procedure MAIN.OV0 and includes another procedure, JOB.INC.

JOB.INC—Used as a temporary storage file.

SUPJOB.INC—Used as a temporary storage file.

Each sub—procedure is loaded into memory when required and then removed following its execution. This technique saves memory, increases flexibility, and is faster in operation.

In the following sections we will look at each of these procedures in detail to understand the mechanism of operation and how they all fit into the conglomerate functionality picture.

82530 Procedure Explanation

COMMON FEATURES

In all the procedures that have been created uniformity has been maintained in certain definitions and functions. In all the procedures the following concepts and notations have been used:

- A message flashes on the screen at the beginning indicating that the procedures are "LOADING....".
- CURHOME and CLEAREOS are invoked after every major operation within the procedure.
- All the titles are in highlight code.
- All the console input requests are in blinking highlight code.
- A message in reverse video code appears on the screen in the end, displaying information on restarting the procedures.
- A BELL will ring if you make a selection that is not on that menu.
- One filename that contains more than one procedure, performs different functions interactively, but by itself performs a specific task.
- All procedures are removed when they are no longer required. This saves user memory space. (This is done by using the REMOVE command and saves user memory).
- If.... THEN....ELSE command statements test the different conditions within a procedure.
- Overlay files explain the main contents of the 82530 Handbook.
- Include files execute the corresponding procedures execution.
- All the local definitions are in the beginning of the procedure.

I²ICE.MAC

This is the marco file that has been created for a Series III/Series IV, with the I²ICE system software, a partial listing of this file is shown. This macro file resides in the same directory as the I²ICE system software. This macro file asks for the drive number that you have assigned for the procedures disk. The drive number is echoed onto the screen by the command "WRITE DEVICE". Depending on the drive selected, literals are defined for "INCLUDE" and "LIST" commands. At the end of the procedure the procedure MAIN.OV0 is invoked using the "INCLUDE" command and then executed. The temporary storage file, SUPJOB.INC, is also invoked using the "INCLUDE" command. The next menu tthat appears on the terminal is the main menu which is invoked by the procedure MAIN.OV0.

MAIN.OV0

This is the main operational procedure. All the other procedures are supervised by this procedure. Every time you are finished with a specific procedure, you are returned to the main menu. From the main menu you can go to the next menu screen or exit to the I²ICE system main interrogation menu if you so desire. Under the MAIN.OV0 filename, there are partitions into two smaller procedures, "W—COM" and "MAIN".

```
 4: *define BOOLEAN flag = FALSE
 5: *.define CHAR device
 6: *.define GLOBAL BYTE drive
 7: *.drive = 0
 8: *.cury = 10t
 9: *.write concat(hibl,'ON WHICH DRIVE IS THE PROCSDISK LOCATED?(0-9)',norm)
11: *.device = ci
12: *.write device
13: *.repeat while not flag
14: *.if device = = '0' then
15: *..drive = 0
16: *..define LITERALLY incc = 'include :f0'
17: *..define LITERALLY lst = 'list :f0'
63: *..end
64: *.flag = true
65: *.define LITERALLY i82530 = 'incc:main.inc nolist'
66: *.end
67: *incc:main.ov0 nolist
68: *main /*execute the procedure*/
69: *incc:supjob.inc nolist
```

280722-47

The procedure "W—COM" contains the LIST and NO-LIST files. If you want to exit from the procedures menu and enter the I²ICE system interrogation menu, then the message "SCC Procedures can be restarted by typing "I82530" appears on the screen in reverse video code.

However, if you make a choice from the main menu then the LIST command writes to SUPJOB.INC. The procedure corresponding to that particular choice is included. The overlay, as well the include portions of the procedure, are included. Then the NOLIST command closes the LIST file.

Note that in the W—COM procedure listing a parameter is being passed into the LIST file, SUBJOB.INC, using the WRITE USING option. The parameter that is being passed depends on the choice you make.

MAIN.INC

The procedure MAIN.INC contains the following two commands.

```
176: *main
177: *incc:supjob.inc nolist
```

280722-64

This procedure does nothing but execute its own procedure MAIN and include the temporary storage file, SUPJOB.INC. Note that SUPJOB.INC contains the include files of your choice. In the preceding example above we have chosen TIMVAL. Hence SUPJOB.INC contains the following two commands:

INCC:TIMVAL.OV0
INCC:TIMVAL.INC

This part of the procedure is the main part of the procedure MAIN.OV0 whose display is shown in Figure 5. In this part your selection appears on the terminal and your choice is echoed onto the screen. If you make a wrong choice, then a "Bell" rings. Depending on the choice made, the corresponding procedure is invoked. As mentioned previously that particular procedure is passed into SUPJOB.INC and executed.

Depending on the selection made by you that particular procedure is invoked.

Example:

If you choose to see the TIME CONSTANT VALUES description, then type in 3. The number 3 appears on the screen and the procedure name, TIMVAL, is passed into the procedure W—COM as a parameter. The procedure W—COM then includes the overlay as well as the include portions of the time constant value procedures using the LIST/NOLIST commands. If you choose the number 8 then you exit to the I²ICE system main menu.

```
126: *.chr = ci
135: *.if chr = = '3' then
136: *..write using ('0') chr
137: *..w__com('timval');return
155: *..if chr = = '8' then
156: *...write using ('0') chr
157: *...w__com('e');return
158: *...else
159: *...bell
160: *...end
161: *..end
```

280722-48

TIMVAL.OV0

All the procedures that contain the 82530 descriptions are built containing the same format except for the pro-

```
 83: *define PROC w_com = DO
 84: *.curhome;cleareos
 85: *.if %0 = = 'e' then
 86: *..curhome;cleareos
 87: *..write concat(revbl,'SCC Procedures can be restarted by typing"i82530")
 88: *..norm
 89: *..lst:supjob.inc
 90: *..nolist
 91: *..lst:job.inc
 92: *..nolist
 93: *..else
 94: *..lst:supjob.inc
 95: *..write using ("incc:",0,".ov0 nolist") %0
 96: *..write using ("incc:",0,".inc nolist") %0
 97: *..curhome;cleareos
 98: *..nolist
 99: *..end
100: *.end
```

280722-49

cedure describing REGISTER BIT FUNCTIONS, PINFUN.OV0. Hence let us consider the TIME CONSTANT VALUES DESCRIPTION procedure as an example to explain the techniques that have been used. The only procedure that is different, the REGISTER BIT FUNCTION procedure, will be explained later in this section.

TIMEVAL.OV0 contains a message "LOADING. . ." that flashes on the terminal when this procedure is being included into the I²ICE system memory. The rest of the procedure contains the information contained in the 82530 Handbook. The WRITE command is used to type all the information on the screen. Towards the end of the procedure is a LIST command file. This LIST file lists the storage file, JOB.INC. The JOB.INC file contains the command that removes the procedure TIMVAL from the I²ICE system memory and invokes the procedure MAIN and also executes it so that you can choose a different option or return to the I²ICE system main menu. Note that I82530 is defined as a literal in the procedure I²ICE.MAC.

```
****************************************
*              82530/82530-6           *
*                                      *
*SERIAL COMMUNICATIONS CONTROLLER (SCC)*
****************************************

     1. PIN DESCRIPTIONS              [DESCRIPTION]
     2. REGISTER FUNCTIONS            [DESCRIPTION]
     3. TIME CONSTANT VALUES          [DESCRIPTION]
     4. DATA ENCODING METHODS         [DESCRIPTION]
     5. REGISTER ADDRESSING           [DESCRIPTION]
     6. SCC PROTOCOLS                 [DESCRIPTION]
     7. REGISTER BIT FUNCTIONS  [INTERROGATE 82530]
     8. EXIT TO I2ICE MAIN MENU

      ------------------------
      PLEASE TYPE YOUR CHOICE
      ------------------------
```

280722-6

Figure 5. Main Menu Register

```
455: *define PROC timval = DO
456: *.write concat(hibl,'LOADING....')
457: *.norm
458: *.curhome;cleareos
459: *.write concat(hi,'TIME CONSTANT VALUES FOR STANDARD BAUD RATES AT BR CLOCK = 3.9936MHz')
460: *.write concat(hi,'*******************************************************************************')
461: *.norm
462: *.write '        BAUD RATE     TIME CONSTANT     ERROR'
463: *.write '          19200           102             -'
464: *.write '           9600           206             -'
465: *.write '           7200           275             0.12%'
466: *.write '           4800           414             -'
467: *.write '           3600           553             0.06%'
468: *.write '           2400           830             -'
469: *.write '           2000           996             0.04%'
470: *.write '           1800           1107            0.03%'
471: *.write '           1200           1662            -'
472: *.write '            600           3326            -'
473: *.write '            300           6654            -'
474: *.write '            150           13310           -'
475: *.write '          134.5           14844           0.0007%'
476: *.write '            110           18151           0.0015%'
477: *.write '             75           26622           -'
478: *.write '             50           39934           -'
479: *.write concat(hibl,'          PLEASE HIT ANY KEY TO RETURN TO SCC MENU')
480: *.norm
481: *.define CHAR yyy = ci
482: *.lst:job.inc
483: *.write 'remove timval'
484: *.write 'i82530'
485: *.nolist
486: *.curhome;cleareos;end
```

280722-7

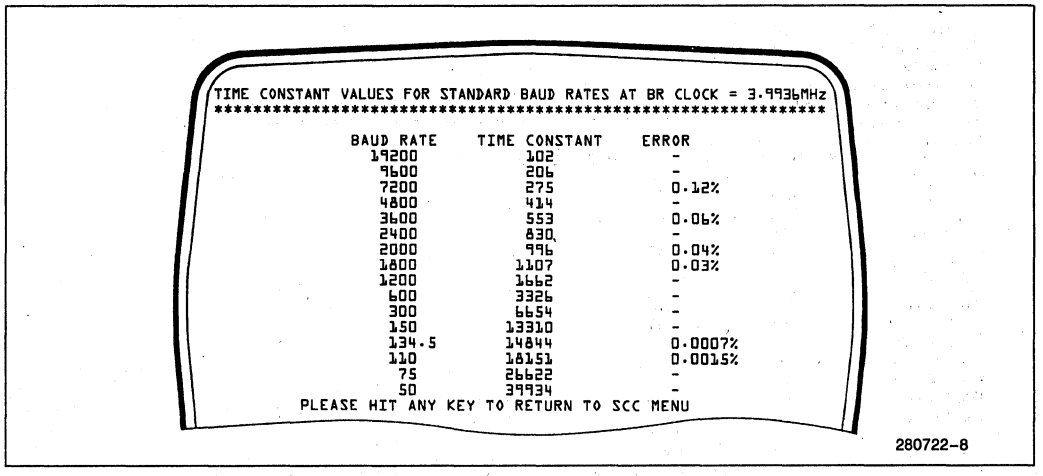


Figure 6. Time Constant Values Description

TIMVAL.INC

The procedure TIMVAL.INC contains the following two commands:

```
493: *timval
494: *incc:job.inc nolist
```

280722-65

This procedure, as well as all the other .INC procedures contains the preceding two commands. The first command executes that specific procedure and the next command includes the storage procedure JOB.INC. As mentioned previously the procedure JOB.INC removes that particular procedure from the I²ICE system memory and also invokes and executes the main procedure.

PINFUN.OV0

This is the procedure that explains the register bit functions of the 82530 Serial Communications Controller. This procedure is very involved and performs all the functions that are required by the 82530 handbook. This procedure, PINFUN.OV0, is divided into a subset of procedures contained within the same file #. The following sections explain the different interactive procedures and their functions, respectively.

GETNUM—This is a procedure that changes the number entered at the console into decimal base. If an inappropriate number is input then the bell rings. The decimal base number is returned to the procedure that calls it. The console inut is echoed onto the screen.

```
736: *define PROC getnum = DO
737: *.define BYTE num
738: *.define CHAR chr
739: *.define CHAR bell = 07h
740: *.num = 0
741: *.repeat
742: *..chr = ci
743: *..if (chr > = '0') and (chr < = '9') then
744: *...num = num*10t + (chr-30h)
745: *...write using ('1,>') chr
746: *...else
747: *...if chr < > 0dh then write using('0,>') bell
748: *....end
749: *...end
750: *..until chr = = 0dh
751: *..end
752: *.return num
753: *.end
```

280722-50

GETP1—This procedure sets the I/O ports for the I²ICE system as well as the starting addresses for the 82530. If you have not mapped the I/O ports, then the method to port is explained on the screen along with menu prompts so that type only the starting address and number of bytes either mapped to ICE memory or USER memory. This procedure also requests your input for the hexadecimal starting address for the 82530.

```
755: *define PROC getp1 = DO
756: *.curhome;cleareos
757: *.define CHAR zzz
758: *.write concat(hi,'MAPIO COMMAND DISPLAYS OR SETS PHYSICAL LOCATION FOR I/O PORTS')
759: *.write concat(norm,'',hibl)
760: *.write using('"                HAVE YOU MAPPED THE I/O PORTS?(Y/N)",>') chrin
761: *.norm
762: *.repeat
763: *..chrin = ci
764: *..if not((chrin = = 'N') or (chrin = = 'n') or (chrin = = 'Y') or (chrin = = 'y')) then
765: *...bell
766: *...endif
767: *..until (chrin = = 'N') or (chrin = = 'n') or (chrin = = 'Y') or (chrin = = 'y')
768: *..end
769: *.if (chrin = = 'N') or (chrin = = 'n') then do
770: *..write using('0') chrin
771: *..curhome;cleareos
772: *..write '          THE I/O PORTS CAN BE MAPPED THE FOLLOWING WAY'
773: *..write ' '
774: *..write '              MAPIO [(partition) <USER or ICE>]'
775: *..write ''
776: *..write '    MAPIO — displays the current map of I/O port address blocks'
777: *..write ''
778: *..write ' partition — is an entry specifying a range of addresses such as:'
779: *..write '              Starting port-address LENGTH number of bytes'
780: *..write ''
781: *..write '    USER — Maps I/O to the user system'
782: *..write ''
783: *..write '    ICE — Maps I/O to the I²ICE probe'
784: *..write ''
785: *..write concat(hibl,'PLEASE TYPE MAPIO STARTING HEXADECIMAL PORT-ADDRESS and <cr>')
786: *..norm
787: *..paddr = gethex
788: *..write 'H'
789: *..write concat(hibl,'    PLEASE TYPE NUMBER OF BYTES and <cr>')
790: *..norm
791: *..paddr1 = gethex
792: *..write 'H'
793: *..write concat(hibl,' PLEASE TYPE "U" FOR USER or "I" FOR ICE')
794: *..norm
795: *..repeat
796: *..zzz = ci
797: *..if not((zzz = = 'U') or (zzz = = 'u') or (zzz = = 'I') or (zzz = = 'i')) then
798: *...bell
799: *...endif
800: *...until (zzz = = 'U') or (zzz = = 'u') or (zzz = = 'I') or (zzz = = 'i')
801: *..end
802: *..if (zzz = = 'I') or (zzz = = 'i') then
803: *..MAPIO PADDR LENGTH PADDR1 ICE
804: *..write using("MAPIO ",0,H," LENGTH ",0,H," ICE "')paddr,paddr1
805: *..write ''
806: *..write concat(hibl,'              PLEASE HIT ANY KEY TO CONTINUE')
807: *..norm
808: *..zzz = ci;else
809: *..if (zzz = = 'U') or (zzz = = 'u') then
810: *...MAPIO PADDR LENGTH PADDR1 USER
811: *...write using("MAPIO ",0,H," LENGTH ",0,H," USER "')paddr,paddr1
812: *...write ''
813: *...write concat(hibl,' PLEASE HIT ANY KEY TO CONTINUE')
814: *...norm
```

280722-9

```
815: *...chrin = ci;endif;end
816: *...end
817: *..end
818: *.curhome;cleareos
819: *.write concat(hibl,'TYPE IN THE HEXADECIMAL ADDRESS FOR THE 82530 and <cr>')
820: *.norm
821: *.wreg = gethex
822: *.write 'H'
823: *.write ''
824: *.end
```

GETP2 — This procedure asks for console input on your choice of whether you want to read a register, or write to one, or change the 82530 port address, or return to the main menu.

```
826: *define PROC getp2 = DO
827: *.curhome;cleareos
828: *.write concat(hi,'          **********')
829: *.write concat(hi,'          *PIN FUNCTION MENU*')
830: *.write concat(hi,'          **********')
831: *.norm
832: *.write ' '
833: *.write ' '
834: *.write '          "R" — READ A REGISTER'
835: *.write '          "W" — WRITE TO A REGISTER'
836: *.write '          "N" — CHANGE 82530 PORT ADDRESS'
837: *.write '          "E" — EXIT TO MAIN MENU'
838: *.write ' '
839: *.write ' '
840: *.write concat(hibl,'          PLEASE TYPE YOUR CHOICE')
841: *.norm
842: *.end
```

GETP3 — This procedure gives the choices of the READ and WRITE REGISTERS that are available for selection and asks for console input.

```
844: *define PROC getp3 = DO
845: *.curhome;cleareos
846: *.write ' '
847: *.write concat(hi,'  ********************************************************** ')
848: *.write concat(hi,'  THE CHOICES OF REGISTERS FOR YOUR SELECTION ARE :')
849: *.write concat(hi,'  **********************************************************')
850: *.norm
851: *.if (chrin = = 'R') or (chrin = = 'r') then
852: *..write '      0. RR0          1. RR1          2. RR2'
853: *..write '      3. RR3          8. RR8         10. RR10'
854: *..write '     12. RR12        13. RR13        15. RR15'
855: *..else
856: *..write '      0. WR0          1. WR1          2. WR2'
857: *..write '      3. WR3          4. WR4          5. WR5'
858: *..write '      6. WR6          7. WR7          8. WR8'
859: *..write '      9. WR9         10. WR10        11. WR11'
860: *..write '     12. WR12        13. WR13        14. WR14'
861: *..write '                     15. WR15  '
862: *..end
863: *.write ' '
864: *.write ' '
865: *.write concat(hibl,'          PLEASE TYPE YOUR CHOICE and <cr>')
866: *.norm
```

280722-10

```
867: *.regnum = getnum
868: *.write 'T'
869: *.write ''
870: *.end
```

280722-66

GETMEN1—This procedure accepts your choice for the pin function menu and, depending on the choice you have made, calls the appropriate procedure. If you want to read or write to a register then the procedure GETP3, which displays the read and write register selection is invoked. On the other hand, if you want to change the 82530 port address then the procedure GETP1, which maps the I/O ports of the I²ICE system, is invoked.

```
872: *define PROC getmen1 = DO
873: *.if check then
874: *..getp1
875: *..check = false
876: *..end
877: *.repeat
878: *.getp2
879: *.chrin = ci
880: *.if (chrin = = 'E') or (chrin = = 'e') then write using ('0') chrin;return end
881: *..if (chrin = = 'R') or (chrin = = 'r') or (chrin = = 'W') or (chrin = = 'w') then
882: *..write using ('0') chrin
883: *..curhome;cleareos
884: *..getp3
885: *..return
886: *..else
887: *..if (chrin = = 'N') or (chrin = = 'n') then
888: *...write using ('0') chrin
889: *...getp1
890: *...else
891: *...if (chrin = = 1bh) then return end
892: *....bell
893: *....end
894: *...end
895: *..end
896: *.end
```

280722-51

W__COM1__ This procedure uses LIST/NOLIST as well as INCLUDE and REMOVE to perform numerous functions. Once the usage of these procedures is complete, all the corresponding procedures are removed using the REMOVE command listed in the procedure JOB.INC and the main menu is invoked. If you choose to read or write to a particular register then that particular procedure containing the register's contents is included. After the register is read or written into, then this procedure removes this particular procedure from the I²ICE system memory and re-invokes the register bit function menu for you to make the next selection.

```
898: *define PROC w__com1 = DO
899: *.curhome;cleareos
900: *..if %0 = = 'e' then
901: *..lst:job.inc
902: *..write 'remove done,gethex,getnum,getmen1,w__com1,getp1,getp2,getp3'
903: *..write 'remove chrin,check,wreg,regnum,paddr,paddr1'
904: *..write 'i82530'
905: *..nolist
906: *..else
907: *..lst:job.inc
908: *..write using ("incc:",0," nolist") %0
909: *..write using ('0') %0
910: *..write using ("remove ",0') %0
911: *..write 'incc:pinfun.inc nolist'
912: *..nolist
913: *..end
914: *.end
```

280722-52

DONE—This procedure is the main body of the file PINFUN.OV0. This is the nucleus which controls the operations of all the other procedures within this filename and inter-acts with them. Depending on your selection, the procedure W__COM1 is executed with the corresponding register value passed as a parameter. If a wrong selection is made, then the message "UNABLE TO READ REGISTER" appears on the screen. Your choice is echoed onto the screen. An inappropriate selection will ring the bell. For example, if you want to read a register "RR0" invoke the procedure W__COM1 and pass the read register procedure RR0 as a parameter as shown in the following listing. Similarly if you want to write to register "WR0", invoke the procedure.

```
916: *define PROC done = DO
917: *.curhome ; cleareos
918: *.getmen1
919: *..if (chrin = = 'E') or (chrin = = 'e') then
920: *..w__com1('e')
921: *..return
922: *..end
923: *..repeat
924: *...if (chrin = = 'R') or (chrin = = 'r') then
925: *...do
926: *....if regnum = = 0 then
927: *....w__com1('rr0') ; return
928: *....end
957: *...getp3
958: *...end
959: *...if (chrin = = 'W') or (chrin = = 'w') then
960: *...do
961: *...if regnum = = 0 then
962: *...w__com1('wr0'); return
963: *...end
1013: *...getp3
1014: *...end
1015: *..end
1016: *.end
```

280722-53

PINFUN.INC

This procedure contains the following two commands:

```
1023: *done
1024: *incc:job.inc nolist
```

280722-67

This procedure executes the main procedure under the filename PINFUN.OV0, DONE, and then includes the temporary storage file, JOB.INC.

RR0

This procedure reads a value from the console which is written into WREG. This procedure also displays this read value in binary form and explains what each of these bits correspond to in an actual 82530.

```
1031: *curhome;cleareos
1032: *write concat(hibl,'LOADING....')
1033: *norm
1034: *define PROC rr0 = DO
1035: *.define CHAR yyy
1036: *.define BYTE temp
1037: *.temp = port(wreg)
1038: *.curhome ;cleareos
1039: *.write concat(hi,' READ REGISTER 0')
1040: *.write concat(hi,' ******************')
```

280722-54

```
1041: *.norm
1042: *.write using ('2c,"=>",2,y,5x,1,y,5x,1,y,5x,1,y,5x,1,y,5x,1,y,5x,1,y') &
1043: *.(temp and 80h) /80h, &
1044: *.(temp and 40h) /40h, &
1045: *.(temp and 20h) /20h, &
1046: *.(temp and 10h) /10h, &
1047: *.(temp and 08h) /08h, &
1048: *.(temp and 04h) /04h, &
1049: *.(temp and 02h) /02h, &
1050: *.(temp and 01h) /01h
1051: *.write concat(norm,'  _____')
1052: *.write concat(norm,'  | D7| D6| D5| D4| D3| D2| D1| D0| ')
1053: *.write concat(norm,'  -----')
1054: *.write '    D7 = BREAK/ABORT'
1055: *.write '       D6 = Tx UNDERRUN/EOM'
1056: *.write '          D5 = CTS'
1057: *.write '             D4 = SYNC/HUNT'
1058: *.write '                D3 = CD'
1059: *.write '                   D2 = Tx BUFFER EMPTY'
1060: *.write '                      D1 = ZERO COUNT'
1061: *.write '                         D0 = Rx CHAR. AVAIL.'
1062: *.write ' '
1063: *.write concat(hibl,' PLEASE HIT ANY KEY TO RETURN')
1064: *.norm
1065: *.yyy=ci
1066: *.end
```

280722-11

WR0

This procedure writes to a register from the console. The eight different bits in the 82530 write register are explained and the console requests for an input from you.

```
1427: *curhome;cleareos
1428: *write concat(hibl,'LOADING....',norm)
1430: *define PROC wr0 = DO
1431: *.define CHAR yyy
1432: *.define BYTE temp
1433: *.curhome;cleareos
1434: *.write concat(hi,'  WRITE REGISTER 0'
1435: *.write concat(hi,'  ******************'
1436: *.write concat(norm,'  _____')
1437: *.write concat(norm,'  | D7| D6| D5| D4| D3| D2| D1| D0| ')
1438: *.write concat(norm,'  -----')
1439: *.write '                D7  D6'
1440: *.write '                    0 0 = NULL CODE '
1441: *.write '                    0 1 = RESET Rx CRC CHECKER'
1442: *.write '                    1 0 = RESET Tx CRC GENERATOR'
1443: *.write '                    1 1 = RESET Tx UNDERRUN/EOM LATCH'
1444: *.write '  D5 D4 D3                                          D2 D1 D0'
1445: *.write '  0  0  0   = NULL CODE                             0  0  0   = 0 or 8'
1446: *.write '  0  0  1   = POINT HIGH REGISTER GROUP             0  0  1   = 1 or 9'
1447: *.write '  0  1  0   = PRESET EXT/STATUS INTERRUPTS          0  1  0   = 2 or 10'
1448: *.write '  0  1  1   = SEND ABORT                           0  1  1   = 3 or 11'
1449: *.write '  1  0  0   = ENABLE INT ON NEXT Rx CHAR.          1  0  0   = 4 or 12'
1450: *.write '  1  0  1   = RESET TxINT PENDING                  1  0  1   = 5 or 13'
1451: *.write '  1  1  0   = ERROR RESET                         1  1  0   = 6 or 14'
1452: *.write '  1  1  1   = RESET HIGHEST IUS                   1  1  1   = 7 or 15'
1453: *.write concat(hibl,'PLEASE TYPE THE VALUE TO BE WRITTEN and <cr>',norm)
1455: *.temp = gethex
```

280722-12

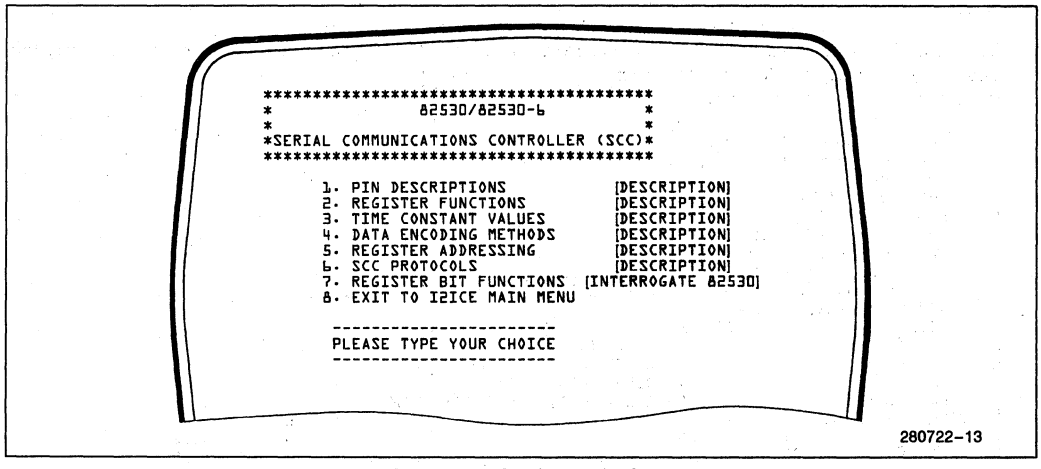
```
1457: *.port(wreg) = 0
1458: *.port(wreg) = temp
1460: *.write concat(hibl,'
1462: *.yyy = ci
1463: *.end
```

280722-68

Example

Let us go through the entire menu with a specific example.

- Select "REGISTER BIT FUNCTION". (Figure 7)
- Select N for "HAVE YOU MAPPED THE I/O PORTS?" (Figure 8)
- Select 0 for "PLEASE TYPE MAPIO HEXADECIMAL PORT ADDRESS and <cr>". (Figure 8)
- Select 40 for "PLEASE TYPE NUMBER OF BYTES and <cr>" (Figure 8)
- Select I for "PLEASE TYPE 'U' FOR USER OR 'I' FOR ICE" (Figure 9)
- Select 0 for "TYPE IN THE HEXADECIMAL ADDRESS FOR THE 82530 and <cr>". (Figure 10)
- Select R for READ A REGISTER (Figure 11)
- Select 0 for "THE CHOICES OF REGISTER FOR YOUR SELECTION ARE:" (Figure 12)
- Select E to "EXIT TO THE MAIN MENU". (Figure 13)
- Select E to "EXIT TO THE I²ICE MAIN MENU". (Figure 14)



280722-13

Figure 7. Main Menu Display

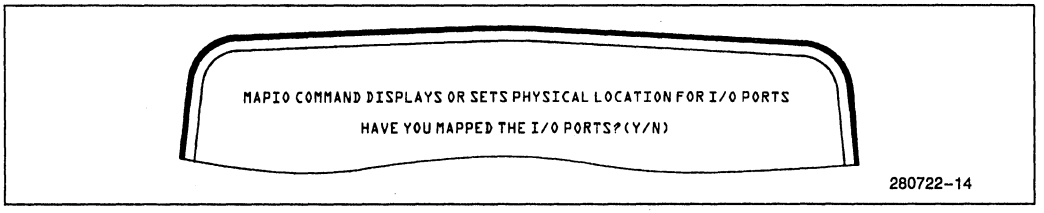


Figure 8. I/O Port Display

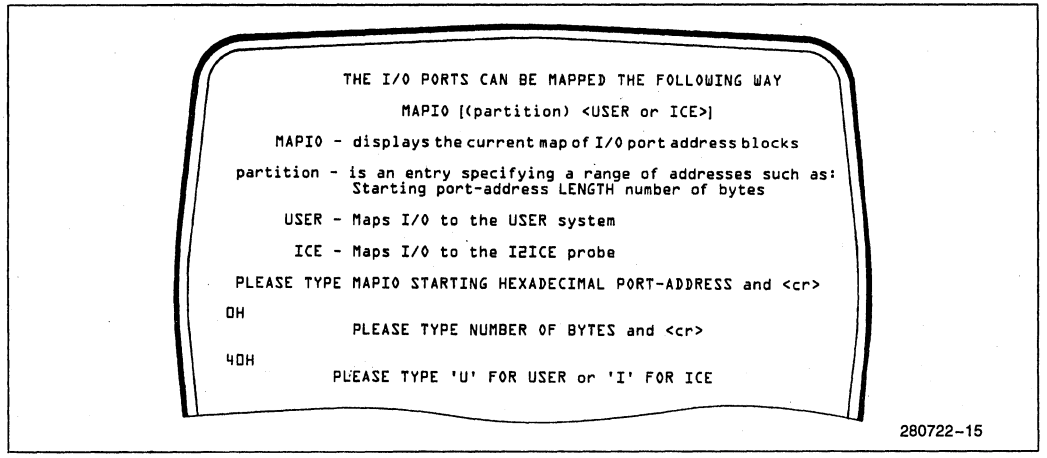


Figure 9. MAPIO Conditions

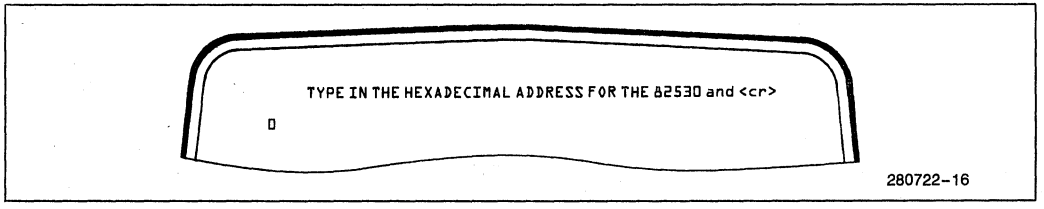


Figure 10. Hexadecimal Port Address

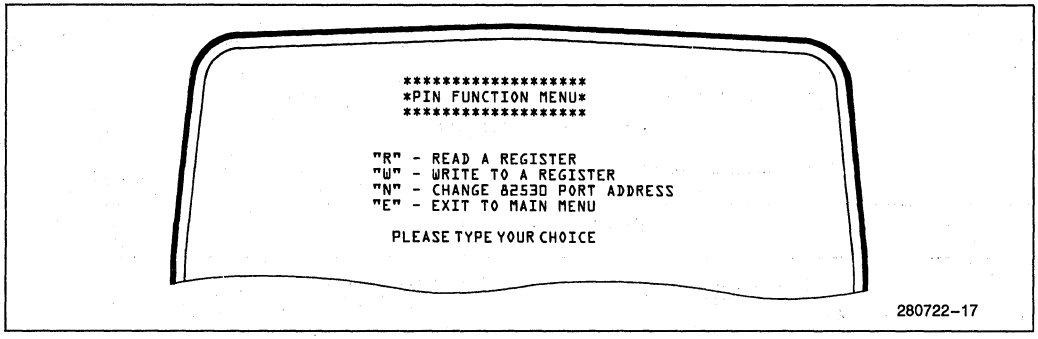


Figure 11. Pin Function Menu

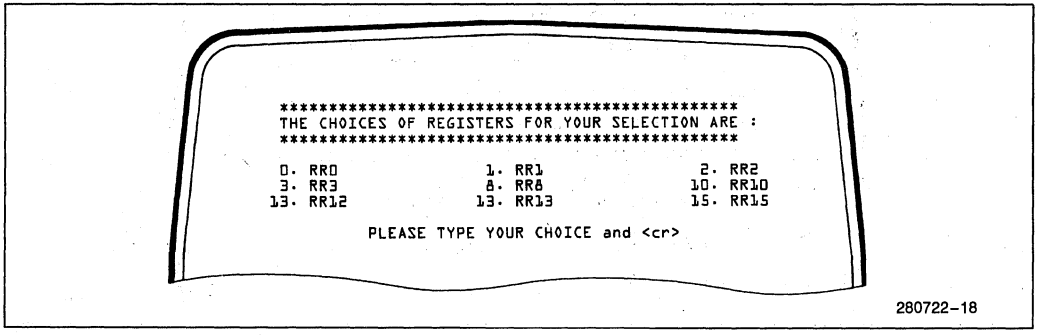


Figure 12. Register Selections

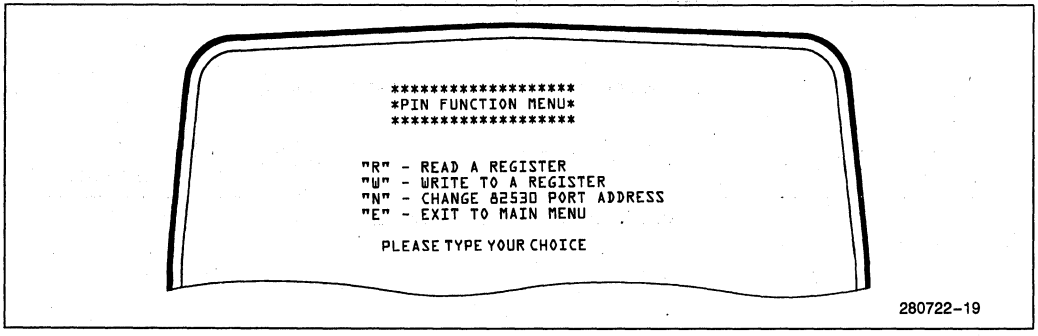


Figure 13. Pin Function Menu

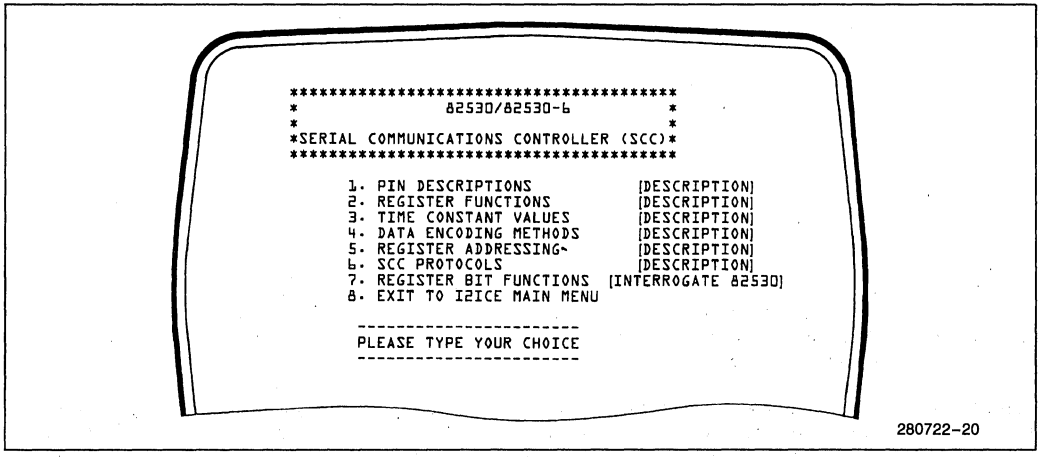


Figure 14. Main Menu

CONCLUSION

The trend in microprocessor instrumentation is toward complete integration of hardware and software design and test components. Debugging systems, system design, and program control utilities will be integrated into one development tool designed to bring users closer to a completely virtual engineering environment.

The I²ICE system combines in-circuit emulation, high-level language software debugging and logic analysis in a system designed to improve productivity in the development and integration of complex microprocessor systems. One of the major features of an I²ICE system is the usage of procedures. Given an I²ICE system, procedures can be written to debug any peripheral chip on the target system by building procedures to simulate the chip and to communicate with it through the I²ICE system. This enhances the debugging capabilities of I²ICE system.

This application note has illustrated the numerous features of procedural implementation using a very simple peripheral component, 8251A, and a complicated component such as the 82530. We have shown that the I²ICE system is able to debug procedures as well as standard microprocessor code. Now you can write your procedures for any peripheral component and hence aid in debugging your design using the I²ICE system. You will spend less time in referring to the data books and more time in your design. This will help to deliver your product on schedule.

APPENDIX A
GLOSSARY

PROCEDURE

A procedure operates like a single command because it enables you to use several commands in a block structure and declare local variables. Also, the procedures can be several nested blocks. The size of procedures is limited only by the amount of memory space available. Defining procedures is like adding commands to the I²ICE language and you can create commands of particular relevance to your debug situation.

Although a debug procedure is not executed until its name is invoked, the I²ICE system checks the syntax when the procedure is defined and determines the type of all referenced objects. Changing the type and/or definition of an object in the procedure before it is executed can cause errors when the procedure is executed.

Procedures can be defined within other procedures. The inner procedure is not visible to the I²ICE system until the outer procedure is executed. Once procedures become visible to the system, they are always global, even when nested inside other procedures. A debug procedure cannot forward reference a debug object. Values can be returned from the procedures, and parameters can be referenced in procedures. The RETURN command is used to return the procedure values to the terminal.

The percent sign (%) signals the I²ICE system to expect an expression.

%NP is a predefined system parameter equal to the number of parameters passed in the debug procedure.

%Number defines a parameter number which selects that parameter from the list following the debug procedure invocation. Numbers range consecutively from 0 to 99.

%(Expression) is used instead of a number but requires parentheses. The expression must evaluate to a number between 0 and 99.

The slash asterisk combination (/*.....*/) defines a comment.

Example:

In the following example we will define an averaging procedure. Data is supplied by the parameter list. This procedure returns the average value of all the parameters.

```
*DEFINE PROC average = DO    /*Define the debug procedure*/
.*DEFINE INTEGER sum = 0
.*DEFINE BYTE I = 0    /*Initialize variables*
.*Count %NP            /*Count is equal to the # of parameters*/
..*sum = sum + %(I)    /*Add I to the sum*/
...*I = I + 1          /*Increment I*/
..*ENDCOUNT            /*Defines the extant of the COUNT loop*/
.*RETURN sum/%NP       /*Return the parameter average*/
.*END
```

Enter the following command to display the debug procedure on the terminal screen.

```
*PROC average                    /*Display the debug procedure definition*/

define proc AVERAGE = do
define integer SUM = 0
define byte I = 0
count %NP
SUM = SUM + %(I)
I = I + 1
endcount
return
SUM/%NP
end
```

The following command executes this debug procedure to find the average of the three numbers.

```
* AVERAGE (4, 5, 21T)                    /*Execute the debug procedure*/
+10
```

But this is only a simple example. I²ICE contains many features which enable us to produce comprehensive procedures. Features such as INCLUDE, LIST, LITERALLY, WRITE, BOOLEAN CONDITIONS and much more.

INCLUDE

The INCLUDE command retrieves a command file from a mass storage device and loads it into memory.

Command files may be created in two ways: by creating a file with the integrated I²ICE screen editor or by saving definitions created during a debug session to a file with the PUT or APPEND command.

Note that INCLUDE has the following restrictions:

- You can nest INCLUDE commands (limited by available memory), but they must be the last item on a line.
- An INCLUDE command cannot appear in block structures (i.e., REPEAT, COUNT, IF, DO/END, or a debug procedure).
- Input cannot originate at the terminal (i.e., INCLUDE :CI:)
- The INCLUDE command must be the last command on the line.

LIST

A LIST file is an I²ICE system utility file. Typically, a list file is used to create a permanent log of a debug session. All interactions between the I²ICE system and the terminal (except edits) are recorded in an open LIST file. Only one list file can be opened at a time. The list files can be closed by issuing the NOLIST command or by terminating a debugging session.

Example:

The following commands open and close a LIST file to record the debug session.

```
*LIST UL16.85
*<Record the debug session>
*NOLIST
```

LITERALLY

LITERALLY definitions are abbreviations for previously defined character strings. LITERALLY definitions save keystrokes and improve clarity. For example, the following definition saves three keystrokes. Once this LITERALLY is defined you can type DEF for DEFINE.

```
*DEFINE LITERALLY DEF = 'DEFINE'
```

These definitions may be saved to disk and auto-reloaded. In addition, an automatic LITERALLY expansion feature can be turned on and off. The examples that follow use this LITERALLY feature extensively.

WRITE

The WRITE command is often used in procedures to add explanatory text to returned values in a more useful form, such as a table.

The WRITE command displays a maximum of 200 bytes of data. Unless specified in the format string by the continuation symbol (&), the information in the write buffer is deleted at the end of every write. If the write-list contains more items than are specified by the format string, the format string is reused from the beginning, until all write-items are displayed according to the format.

Example:

The following example shows the WRITE USING option. The procedure SQUAREIT squares a number specified when the procedure is called (%0).

```
*DEFINE    PROC    squareit = DO
.*WRITE      USING( "The square of ",X,T,0,X,"is",X,T,0,')%0,%0*%0
.*END
```

Call the procedure and specify the number to be squared.

```
*squareit(7)
The square of 7 is 49
```

ATTRIBUTE CODES

Attribute codes are used to add clarity or distinction to text written to the screen. These attribute codes are used with the CONCAT command. The CONCAT command builds strings by concatenating all or part of old strings to form a new string. Placing a CONCAT function inside of a debug procedure saves the construction and prints it when the procedure is executed. The various codes that are available on a host system are:

Attribute Codes	Series III/Series IV	IBM-PC
Blinking Code	82II	5
Reverse Video	90H	7
Highlight Code	81H	1
Blinking/Reverse Video	92H	7;5
Blinking/Highlight	83H	1;5
Normal	80GH	0

Example:

In this example "Hello!" is written in blinking code and the screen returns to the normal attribute code immediately afterwards. The following steps perform these functions. Note that the rest of the screen will stay in the specified attribute mode until the mode is changed or turned off.

SeriesIII/SeriesIV

```
*define global CHAR esc = 1bh
*define global CHAR blink = concat (esc, 'L',82H)
*define global CHAR norm = concat(esc, 'L',80H)
*write concat(blink, 'Hello', norm) /*BLINKING CODE*/
 Hello!
```

IBM PC XT/AT

```
*define global CHAR esc = 1bh
*define global CHAR blink = concat (esc,4eh,esc,"[0m',esc,"[5m')
*define global CHAR norm = concat(esc,"[0m",esc,4eh)
*write concat(blink, "Hello! "',norm) /*BLINKING CODE*/
Hello!
```

BOOLEAN-CONDITION

A Boolean condition is either a value of type BOOLEAN (TRUE or FALSE) or an expression that uses one of the following relational operators:

= = equal to
> greater than
< less than
> = greater than or equal to
< = less than or equal to
< > not equal to

CI-

The CI (console input) function enables a debug procedure to read one character from the system terminal. The procedure pauses until the character is entered. No prompt is displayed while the system is waiting for the CI character, and the entered character is not echoed to the screen. No carriage return is required after the character has been keyed in.

IF

An IF command conditionally executes a command or group of commands. Debug objects are local only in memory type definitions and DO/END blocks. Literals and debug procedures are always global.

Syntax:

If boolean-condition THEN
[I²ICE commands]
[ELSE[I²ICE commands]]
END[IF]

APPENDIX B
PROCEDURES LISTINGS

```
                                    *I2ICE.MAC*
                                    ***********

define BOOLEAN flag = FALSE
define CnAR device
define GLOBAL BYTE drive
drive = 0
cury = 10t
write concat(1bh,'L',83h), '          ON WHICH DRIVE IS THE PROC:DISK LOCATED?(0 - 9) '
write concat(1bh,'L',80h),''
device = ci
write device
repeat while not flag
if device=='0' then
drive=0
define LITERALLY incc='include :f0'
define LITERALLY lst = 'list :f0'
else
if device=='1' then
drive=1
define LITERALLY incc='include :f1'
define LITERALLY lst = 'list :f1'
else
if device=='2' then
drive=2
define LITERALLY incc='include :f2'
define LITERALLY lst = 'list :f2'
else
if device=='3' then
drive=3
define LITERALLY incc='include :f3'
define LITERALLY lst = 'list :f3'
else
if device=='4' then
drive=4
define LITERALLY incc='include :f4'
define LITERALLY lst = 'list :f4'
else
if device=='5' then
drive=5
define LITERALLY incc='include :f5'
define LITERALLY lst = 'list :f5'
else
if device=='6' then
drive=6
define LITERALLY incc = ' include :f6'
define LITERALLY lst = 'list :f6'
else
if device=='7' then
drive=7
define LITERALLY incc='include :f7'
define LITERALLY lst = 'list :f7'
else
if device=='8' then
drive=8
define LITERALLY incc='include :f8'
define LITERALLY lst = 'list :f8'
else
if device=='9' then
drive=9
define LITERALLY incc='include :f9'
define LITERALLY lst = 'list :f9'
end;end;end;end;end;end;end;end;end;endif
flag = true
define LITERALLY i82530 = 'incc:main.inc nolist'
end
incc:main.ov0 nolist
main /*execute the procedure*/
incc:supjob.inc nolist



                                    *MAIN.OV0*
                                    **********

cury = 20t
write concat(1bh,'L',83h),'LOADING....'
write concat(1bh,'L',80h),''
define PROC main = DO
define CnAR bell = 07h
define CHAR chr
```

280722-21

```
define PROC w_com = DO
curhome;cleareos
if %0 == '0' then
curhome;cleareos
write concat(1bh,'L',90h),'SCC Procedures can be restarted by typing "i82530"'
write concat(1bh,'L',80h),' '
lst:supjob.inc
nolist
lst:job.inc
nolist
else
lst:supjob.inc
write using ('"incc:",0,".ov0 nolist"') %0
write using ('"incc:",0,".inc nolist"') %0
curhome;cleareos
nolist
end
end

curhome;cleareos
write ' '
write concat(1bh,'L',81h),'              ****************************************'
write concat(1bh,'L',81h),'              *                82530/82530-6         *'
write concat(1bh,'L',81h),'              *                                      *'
write concat(1bh,'L',81h),'              *SERIAL COMMUNICATIONS CONTROLLER(SCC)*'
write concat(1bh,'L',81h),'              ****************************************'
write concat(1bh,'L',80h),' '
write ' '
write '                      1.  PIN DESCRIPTIONS            [DESCRIPTION]'
write '                      2.  REGISTER FUNCTIONS          [DESCRIPTION]'
write '                      3.  TIME CONSTANT VALUES        [DESCRIPTION]'
write '                      4.  DATA ENCODING METHODS       [DESCRIPTION]'
write '                      5.  REGISTER ADDRESSING         [DESCRIPTION]'
write '                      6.  SCC PROTOCOLS               [DESCRIPTION]'
write '                      7.  REGISTER BIT FUNCTIONS   [INTERROGATE 82530]'
write '                      8.  EXIT TO I2ICE MAIN MENU  '
write ' '
write ' '
write concat(1bh,'L',81h),'                          ------------------------'
write concat(1bh,'L',83h),'                          PLEASE TYPE YOUR CHOICE'
write concat(1bh,'L',81h),'                          ------------------------'
write concat(1bh,'L',80h),' '
repeat
chr = ci
if chr=='1' then
write using ('0') chr
w_com('pindes');return
else
if chr == '2' then
write using ('0') chr
w_com('regdes');return
else
if chr == '3' then
write using ('0') chr
w_com('timval');return
else
if chr == '4' then
write using ('0') chr
w_com('datenc');return
else
if chr == '5' then
write using ('0') chr
w_com('regadd');return
else
if chr== '6' then
write using ('0') chr
w_com('prtcol');return
else
if chr == '7' then
write using ('0') chr
w_com('pinfun');return
else
if chr == '8' then
write using ('0') chr
w_com('e');return
else
bell
end
end
end
end
end
```

280722-22

```
end
end
end
end
end


                                    *MAIN.INC*
                                    **********

main
incc:supjob.inc nolist


                                    *PINDES.OVO*
                                    ************

curhome;cleareos
write concat(1bh,'L',83h),'LOADING....'
write concat(1bh,'L',80h),''
define* CHAR ppp
define PROC pinds = DO
curhome;cleareos
write concat(1bh,'L',81h),'*82530/82530-6 SERIAL COMMUNICATIONS CONTROLLER (SCC)*'
write concat(1bh,'L',81h),'************************************************** '
write concat(1bh,'L',81h),'              PIN DESCRIPTION'
write concat(1bh,'L',81h),'              ###############'
write concat(1bh,'L',81h),' SYMBOL   PIN NO.   TYPE   NAME AND FUNCTION'
write concat(1bh,'L',81h),' @@@@@@   @@@@@@@   @@@@   @@@@@@@@@@@@@@@@@@'
write concat(1bh,'L',80h),' '
end
define PROC pindes = DO
pinds
write '      DB0       40       I/O    DATA BUS:The Data Bus lines are '
write '      DB1        1       I/O    bi-directional three-state lines'
write '      DB2       39       I/O    which interface with the systems'
write '      DB3        2       I/O    Data Bus. These lines carry data'
write '      DB4       38       I/O    and commands to and from the SCC.'
write '      DB5        3       I/O'
write '      DB6       37       I/O'
write '      DB7        4       I/O'
write ' '
write '      INT        5        O     INTERRUPT REQUEST:The interrupt signal'
write '                                 is activated when the SCC requests an'
write '                                 interrupt. It is an open drain output.'
write concat(1bh,'1',83h),'              PLEASE HIT ANY KEY TO CONTINUE'
write concat(1bh,'L',80h),''
ppp = ci
pinds
write '      IEO        6        O     INTERRUPT ENABLE OUT:is nigh only if IEI'
write '                                 is High and the CPU is not servicing an'
write '                                 SCC interrupt or the SCC is not requesting'
write '                                 an interrupt.'
write ' '
write '      IEI        7        I     INTERRUPT ENABLE IN:is used with IEO to form'
write '                                 an interrupt daisy chain when there is more'
write '                                 than one interrupt-driven device.'
write '     ____'
write '     INTA       8        I     INTERRUPT ACKNOWLEDGE:indicates an active '
write '                                 interrupt acknowledge cycle. During this '
write '                                 cycle, the SCC interrupt daisy chain settles.'
write concat(1bh,'L',83h),'              PLEASE nIT ANY KEY TO CONTINUE'
write concat(1bh,'L',80h),''
ppp = ci
pinds
write '      VCC        9              POWER:5V Power supply'
write '  ___ ___'
write '  RDYa/REQa  10         O      READY/REQUEST:(Output, open-drain when '
write '  ___ ___                      programmed for a Ready function, driven'
write '  RDYb/REQb             O      nigh or Low when programmed for a Request func.'
write '     ____'
write '     SYNCa      11       I/O    SYNCnRONIZATION: These pins can attract'
write '                                 either as inputs, outputs or part of the'
write '                                 crystal oscillator circuit.'
write '     ____'
write '     RTxCa      12       O      RECEIVE/TRANSMIT CLOCKS: These pins can be'
write '                                 be programmed in several different modes of'
write '     ____                        operation.  '
write '     RTxCb'
write concat(1bh,'L',83h),'              PLEASE nIT ANY KEY TO CONTINUE'
write concat(1bh,'L',80h),''
```

280722-23

```
247:  ppp = ci
248:  pinds
249:  write '       ____'
250:  write '      TRxCa       14        I/O    TRANSMIT/RECEIVE CLOCKS:These pins can be'
251:  write '      ____                         programmed in several different modes of'
252:  write '      TRxCb       26        I/O    operation. The receive clock or the transmit'
253:  write '                                   clock in the input mode or the the output of'
254:  write '                                   the Digital Phase Locked Loop, the crystal'
255:  write '                                   oscillator, the baud rate generator, or the'
256:  write '                                   transmit clock in the output mode may be'
257:  write '                                   supplied.'
258:  write concat(1bh,'l',83h),'                        PLEASE HIT ANY KEY TO CONTINUE'
259:  write concat(1bh,'L',80h),''
260:  ppp = ci
261:  end
262:  define PROC pndsad  = DO
263:  pinds
264:  write '       TxDa       15          O    TRANSMIT DATA:These output signals transmit'
265:  write '       TxDa       25          O    serial data at standard TTL levels.'
266:  write '      ___ ____'
267:  write '      DTRaREQa    16          O    DATA TERMINAL READY/REQUEST:These outputs'
268:  write '      ___ ___                      follow the state programmed into the DTR bit'
269:  write '      DTRbREQb    24          O    They can also be used as general purpose'
270:  write '                                   outputs or as Request lines for a DMA contr.'
271:  write '       ____'
272:  write '       RTSa       17          O    REQUEST TO SEND:When the RTS bit in write'
273:  write '       ___                         Register 5 is set, the signal goes low. '
274:  write '       RTSb       23          O    When the RTS bit is reset in the Async mode'
275:  write '                                   and Auto enable is on, the signal goes high'
276:  write '                                   after the transmitter is empty.'
277:  write concat(1bh,'l',83h),'                        PLEASE HIT ANY KEY TO CONTINUE'
278:  write concat(1bh,'L',80h),''
279:  ppp = ci
280:  pinds
281:  write '       RxDa       13          I    RECEIVE DATA:These lines receive serial'
282:  write '       RxDb       27          I    data at standard TTL Levels'
283:  write '       ___'
284:  write '       CTSa       18          I    CLEAR TO SEND:If these pins are programmed'
285:  write '       ___                         as Auto Enables, a Low on the inputs enables'
286:  write '       CTSb       22          I    the respevtive transmitters. If not'
287:  write '                                   programmed as Auto Enables, they may be used'
288:  write '                                   as general purpose inputs.'
289:  write '       __'
290:  write '       CDa        19          I    CARRIER DETECT:These pins function as'
291:  write '       __                          receiver enables if they are programmed'
292:  write '       CDb        21          I    for Auto Enables; otherwise they may be used'
293:  write '                                   as general purpose input pins.'
294:  write concat(1bh,'L',83h),'                        PLEASE HIT ANY KEY TO CONTINUE'
295:  write concat(1bh,'L',80h),''
296:  ppp = ci
297:  pinds
298:  write '       CLK        20          I    CLOCK:This is the system SCC clock used to'
299:  write '                                   synchronize internal signals.(TTL level)'
300:  write '         _'
301:  write '       D/C        32          I    DATA/COMMAND SELECT:This signal defines the'
302:  write '                                   type of information transferred to or from'
303:  write '                                   the SCC. High means data is transferred, a'
304:  write '                                   Low indicates a command.'
305:  write '       __'
306:  write '       CS         33          I    CHIP SELECT:This signal selects the SCC for '
307:  write '                                   a read or write operation.'
308:  write concat(1bh,'L',83h),'                        PLEASE HIT ANY KEY TO CONTINUE'
309:  write concat(1bh,'L',80h),''
310:  ppp = ci
311:  pinds
312:  write '         _'
313:  write '       A/B        34          I    CHANNEL A/CHANNEL B SELECT:This signal '
314:  write '                                   selects the channel in which the read or'
315:  write '                                   write operation occurs.'
316:  write '        __'
317:  write '       WR         35          I    WRITE:When the SCC is selected this signal'
318:  write '                                   indicates a write operation.'
319:  write '        __'
320:  write '       RD         36          I    READ:This signal indicates a read operation and'
321:  write '                                   when the SCC is selected, enables its bus drivers.'
322:  write ' '
323:  write '       GND        31               GROUND'
324:  write concat(1bh,'l',83h),'                  PLEASE HIT ANY KEY TO RETURN TO SCC MENU'
325:  write concat(1bh,'L',80h),' '
326:  ppp = ci
327:  lst:job.inc
328:  write 'remove ppp'
```

280722-24

```
write 'remove pindes,pndsad,pinds'
write 'i82530'
nolist
end


                                    *PINDES.INC*
                                    ************

pindes
pndsad
incc:job.inc nolist


                                    *REGDES.OV0*
                                    ************

curhome;cleareos
write concat(lbh,'L',83h),'LOADING....'
write concat(lbh,'L',80h),''
define PROC regdes = DO
repeat;curhome;cleareos
write concat(lbh,'L',81h),'                    ************************************'
write concat(lbh,'L',81h),'                    *READ AND WRITE REGISTER DESCRIPTION*'
write concat(lbh,'L',81h),'                    ************************************'
write concat(lbh,'L',80h),' '
write '                    1. READ REGISTER DESCRIPTION'
write '                    2. WRITE REGISTER DESCRIPTION'
write '                    3. EXIT TO SCC MAIN MENU'
write ' '
write concat(lbh,'L',83h),'                                  PLEASE TYPE YOUR CHOICE'
write concat(lbh,'L',80h),' '
define CHAR yyy = ci
until yyy == '3'
if yyy =='1' then
curhome;cleareos
write concat(lbh,'L',81h),'READ REGISTER DESCRIPTION          '
write concat(lbh,'L',80h),' '
write 'RR0   Transmit/Receive buffer status and External status'
write ' '
write 'RR1   Special Receive Condition status'
write ' '
write 'RR2   Modified Interrupt vector (Channel B only)'
write '      Unmodified Interrupt (Channel A only)'
write ' '
write 'RR3   Interrupt Pending bits (Channel A only)'
write ' '
write 'RR8   Receive buffer'
write ' '
write 'RR10  Miscellaneous status'
write ' '
write 'RR12  Lower byte of baud rate generator time constant'
write ' '
write 'RR13  Upper byte of baud rate generator time constant'
write ' '
write 'RR15  External/Status interrupt information'
write concat(lbh,'L',83h),'                                  PLEASE HIT ANY KEY TO RETURN'
write concat(lbh,'L',80h),''
define CHAR zzz = ci
else
if yyy == '2' then
curhome;cleareos
write concat(lbh,'L',81h),'WRITE REGISTER DESCRIPTION'
write concat(lbh,'L',80h),' '
write 'WR0   CRC initialize, initialization commands for the various modes,'
write '      shift right/shift left command.'
write ' '
write 'WR1   Transmit/Receive interrupt and data transfer mode definition'
write ' '
write 'WR2   Interrupt vector (accessed through either channel)'
write ' '
write 'WR3   Receive parameters and control'
write ' '
write 'WR4   Transmit/Receive miscellaneous parameters and modes'
write ' '
write 'WR5   Transmit parameters and controls'
write ''
write 'WR6   Sync characters or SDLC address field'
write concat(lbh,'L',83h),' PLEASE HIT ANY KEY TO CONTINUE'
write concat(lbh,'L',80h),''
define CHAR zzz = ci
```

280722-25

```
411:  curhome;cleareos
412:  write 'WR7   Sync characters or SDLC address field'
413:  write ''
414:  write 'WR8   Transmit buffer'
415:  write ' '
416:  write 'WR9   Master interrupt control and reset (accessed through either'
417:  write '      channel)'
418:  write ' '
419:  write 'WR10  Miscellaneous transmitter/receiver control bits'
420:  write ' '
421:  write 'WR11  Clock mode control'
422:  write ' '
423:  write 'WR12  Lower Byte of baud rate generator time constant'
424:  write ' '
425:  write 'WR13  Upper Byte of baud rate generator time constant'
426:  write ' '
427:  write 'WR14  Miscellaneous control bits'
428:  write ' '
429:  write 'WR15  External/Status interrupt control'
430:  write concat(1bh,'L',83h),'                              PLEASE HIT ANY KEY TO RETURN'
431:  write concat(1bh,'L',80h), ' '
432:  define CHAR zzz = ci
433:  end
434:  end
435:  end
436:  lst:job.inc
437:  write 'remove regdes'
438:  write 'i82530'
439:  nolist
440:  end
441:
442:
443:
444:                                *REGDES.INC*
445:                                ************
446:
447:  regdes
448:  incc:job.inc nolist
449:
450:
451:
452:                                *TIMVAL.OV0*
453:                                ************
454:
455:  define PROC timval = DO
456:  write concat(1bh,'L',83h),'LOADING....'
457:  write concat(1bh,'L',80h),''
458:  curhome;cleareos
459:  write concat(1bh,'L',81h),'TIME CONSTANT VALUES FOR STANDARD BAUD RATES
460:  write concat(1bh,'L',81h),'*************************************************
461:  write concat(1bh,'L',80h), ' '
462:  write '                BAUD RATE     TIME CONSTANT     ERROR'
463:  write '                  19200           102             -'
464:  write '                   9600           206             -'
465:  write '                   7200           275             0.12%'
466:  write '                   4800           414             -'
467:  write '                   3600           553             0.06%'
468:  write '                   2400           830             -'
469:  write '                   2000           996             0.04%'
470:  write '                   1800          1107             0.03%'
471:  write '                   1200          1662             -'
472:  write '                    600          3326             -'
473:  write '                    300          6654             -'
474:  write '                    150         13310             -'
475:  write '                  134.5         14844             0.0007%'
476:  write '                    110         18151             0.0015%'
477:  write '                     75         26622             -'
478:  write '                     50         39934             -'
479:  write concat(1bh,'L',83h),'           PLEASE HIT ANY KEY TO RETURN TO SCC MENU'
480:  write concat(1bh,'L',80h), ' '
481:  define CHAR yyy = ci
482:  lst:job.inc
483:  write 'remove timval'
484:  write 'i82530'
485:  nolist
486:  curhome;cleareos;end
487:
488:
489:
490:                                *TIMVAL.INC*
491:                                ************
492:
```

280722-26

```
493:   timval
494:   incc:job.inc nolist
495:
496:
497:
498:                                    *DATENC.OV0*
499:                                    ************
500:
501:   curhome;cleareos
502:   write concat(1bh,'L',83h),'LOADING....'
503:   write concat(1bh,'L',80h),''
504:   define PROC datenc = DO
505:   repeat
506:   curhome;cleareos
507:   write concat(1bh,'L',81h),'                        **********************'
508:   write concat(1bh,'L',81h),'                        *DATA ENCODING METHODS* '
509:   write concat(1bh,'L',81h),'                        **********************'
510:   write concat(1bh,'L',80h), ' '
511:   write ' '
512:   write ' '
513:   write '                              1. DATA TABLE '
514:   write '                              2. TIMING DIAGRAM'
515:   write '                              3. EXIT TO SCC MAIN MENU'
516:   write ' '
517:   write ' '
518:   write ' '
519:   write ' '
520:   write ' '
521:   write ' '
522:   write ' '
523:   write concat(1bh,'L',83h),'                        PLEASE TYPE YOUR CnOICE'
524:   write concat(1bh,'L',80h),''
525:   define CnAR zzz = ci
526:   until zzz=='3'
527:   if zzz=='1' then
528:   curhome;cleareos
529:   write concat(1bh,'L',81h),'MODE          SYSTEM       SYSTEM CLOCK    SERIAL      CONDITIONS'
530:   write concat(1bh,'L',81h),'              CLOCK        SERIAL CLOCK    BIT RATE    '
531:   write concat(1bh,'L',81h),'****          ******       ************    ********    **********'
532:   write concat(1bh,'L',80h), ' '
533:   write 'Serial        4Mnz             4            1 Mbps       Serial clocks'
534:   write 'clocks                                                    synchronized with'
535:   write 'generated                                                 system clock'
536:   write 'externally    6Mnz             4            1.5 Mbps     Same as above'
537:   write ' '
538:   write '              4Mnz             4.5          880 Kbps     Serial clocks and'
539:   write '                                                          system clock async'
540:   write '              6MHz             4.5          1.3 Mbps     Same as above'
541:   write ''
542:   write 'Self-clocked '
543:   write 'Operation'
544:   write 'NRZI          4MHz             32           125 Kbps'
545:   write '              6Mnz             32           187 Kbps'
546:   write ' '
547:   write 'FM            4Mnz             16           250 Kbps'
548:   write '              6Mnz             16           375 Kbps'
549:   write concat(1bh,'1',83h), '                          PLEASE HIT ANY KEY TO RETURN'
550:   write concat(1bh,'L',80h),''
551:   define CHAR yyy=ci
552:   else if zzz=='2' then
553:   curhome;cleareos
554:   write concat(1bh,'L',81h),'                          TIMING DIAGRAM '
555:   write concat(1bh,'L',81h),'                          ************** '
556:   write concat(1bh,'L',80h), ' '
557:   write ' DATA    1   ¶   1   ¶   0   ¶   0   ¶   1   ¶   0   '
558:   write '          ___________                 _______             HIGH = 1'
559:   write '  NRZ                ¶               ¶       ¶           '
560:   write '                      ----------------        ---------LOW = 0'
561:   write '  NRZ1_______________             _______________         NO CHANGE=1'
562:   write '                      ¶           ¶               ¶'
563:   write '                      ----------                  ---------CHANGE = 0'
564:   write '  FM1  _____  _____   _________   _____                   BIT CENTER'
565:   write '     ¶    ¶   ¶   ¶   ¶       ¶   ¶   ¶   ¶           ¶TRANSITION'
566:   write '     -----    -----   ----------      ----    ---------TRANSITION=1'
567:   write '                                                        NO TRANSITION=0'
568:   write '  FM0 ________  ______  _____   _____                  NO TRANSITION=1'
569:   write '     ¶        ¶       ¶   ¶   ¶   ¶   ¶       ¶   ¶   ¶TRANSITION=0'
570:   write '     ---------  -----   -----   --------  -----'
571:   write 'MAN- _____  _____       ______  ________        _____ LOW-HIGH=0'
572:   write 'CnE-     ¶  ¶   ¶       ¶   ¶   ¶       ¶       ¶     nIGn-LOW=1'
573:   write 'STER      -----  ---------   -----   ---------'
574:   write concat(1bh,'1',83h), '                PLEASE nIT ANY KEY TO RETURN'
```

280722-27

```
575:   write concat(lbh,'L',80h),''
576:   define CHAR yyy=ci
577:   end
578:   end
579:   end
580:   lst:job.inc
581:   write 'remove datenc'
582:   write 'i82530'
583:   nolist
584:   curhome;clearecs;end
585:
586:
587:
588:                                  *DATENC.INC*
589:                                  ************
590:
591:   datenc
592:   incc:job.inc nolist
593:
594:
595:
596:                                  *REGADD.OV0*
597:                                  ************
598:
599:   curhome ; clearecs
600:   write concat(lbh,'L',93h),'LOADING....'
601:   write concat(lbh,'L',90h),''
602:   define PROC regadd = DO
603:   curhome ; clearecs
604:   write concat(lbh,'L',81h),'D/C *POINT HIGH* CODE    D2    D1    D0     WRITE       READ'
605:   write concat(lbh,'L',81h),'          IN WR0              IN WR0          REGISTER   REGISTER'
606:   write concat(lbh,'L',80h), ' '
607:   write '  High        Either way    X     X     X     Data       Data'
608:   write '  Low         Not true      0     0     0      0          0'
609:   write '  Low         Not true      0     0     1      1          1'
610:   write '  Low         Not true      0     1     0      2          2'
611:   write '  Low         Not true      0     1     1      3          3'
612:   write '  Low         Not true      1     0     0      4         (0)'
613:   write '  Low         Not true      1     0     1      5         (1)'
614:   write '  Low         Not true      1     1     0      6         (2)'
615:   write '  Low         True          1     1     1      7         (3)'
616:   write '  Low         True          0     0     0     Data       Data'
617:   write '  Low         True          0     0     1      9          -'
618:   write '  Low         True          0     1     0      10         10'
619:   write '  Low         True          0     1     1      11        (15)'
620:   write '  Low         True          1     0     0      12         12'
621:   write '  Low         True          1     0     1      13         13 '
622:   write '  Low         True          1     1     0      14        (10)'
623:   write '  Low         True          1     1     1      15         15'
624:   write concat(lbh,'L',83h),'          PLEASE HIT ANY KEY TO RETURN TO SCC MENU'
625:   write concat(lbh,'L',80h), ' '
626:   define CHAR yyy = ci
627:   lst:job.inc
628:   write 'remove regadd'
629:   write 'i82530'
630:   nolist
631:   curhome;clearecs
632:   end
633:
634:
635:
636:                                  *REGADD.INC*
637:                                  ************
638:
639:   regadd
640:   incc:job.inc nolist
641:
642:
643:
644:                                  *PRTCOL.OV0*
645:                                  ************
646:
647:   curhome;clearecs
648:   write concat(lbh,'L',83h),'LOADING....'
649:   write concat(lbh,'L',80h),''
650:   define PROC prtcol= DO
651:   curhome;clearecs
652:   write concat(lbh,'L',81h),'                                    SCC PROTOCOLS'
653:   write concat(lbh,'L',80h),'          Start   Parity'
654:   write '_______  ____________  ____  ____  ____  ____  ____  ____  ____  _______'
655:   write 'Marking¶  ¶Data ¶ ¶    ¶  ¶Data ¶ ¶    ¶  ¶Data ¶  ¶    ¶  ¶Marking '
656:   write 'Line   ------------    ------------    ------------    Line'
```

280722-28

```
657:  write '                          "ASYNCaRONOUS"'
658:  write '  ________________________/ /_____________________________________'
659:  write '  ¶SYNC   ¶ DATA  ¶      "MONOSYNC"       ¶ DATA   ¶  CRC1  ¶CRC2 ¶'
660:  write '  ------------------------/ /-------------------------------------'
661:  write '  ________________________/ /_____________________________________'
662:  write '¶ ¶ SYNC  ¶ DATA  ¶       "BISYNC"        ¶ DATA   ¶  CRC1  ¶CRC2 ¶'
663:  write '--------------------------/ /-------------------------------------'
664:  write '          Signal          '
665:  write '          ________________/ /_____________________________________'
666:  write '          ¶ DATA  ¶     "EXTERNAL SYNC"   ¶ DATA   ¶  CRC1  ¶CRC2 ¶'
667:  write '          ----------------/ /-------------------------------------'
668:  write '  ________________________/ /_____________________________________'
669:  write '¶ ¶ADDR   ¶                INFORMATION                ¶ CRC1   ¶CRC2 ¶    ¶'
670:  write '--------------------------/ /-------------------------------------'
671:  write 'FLAG                   "SDLC/HDLC/X.25"'
672:  write concat(1bh,'L',83h),'            PLEASE HIT ANY KEY TO RETURN TO SCC MENU'
673:  write concat(1bh,'L',80h),''
674:  define CHAR yyy = ci
675:  lst:job.inc
676:  write 'remove prtcol'
677:  write '18253()'
678:  nolist
679:  curhome;cleareos
680:  end
681:
682:
683:
684:                                  "PRTCOL.INC"
685:                                  ************
686:
687:  prtcol
688:  incc:job.inc nolist
689:
690:
691:
692:                                  "PINFUN.OV()"
693:                                  ************
694:
695:  curhome;cleareos
696:  write concat(1bh,'L',83h),'LOADING ....'
697:  write concat(1bh,'L',80h), ' '
698:  curhome;cleareos
699:
700:  define WORD paddr
701:  define WORD paddr1
702:  define CHAR chrin
703:  define BOOLEAN check = true
704:  define WORD wreg
705:  define BYTE regnum
706:  define CHAR bell = 07h
707:
708:  define PROC gethex = DO
709:  define WORD num
710:  define CHAR chr
711:  num = 0
712:  repeat
713:  chr = ci
714:  if (chr >= '0') and (chr <='9') then
715:  num = num*10h + (chr-30h)
716:  write using ('1,>') chr
717:  else
718:  if (chr >= 'A') and (chr <= 'F') then
719:  num = num*10h + (chr-37h)
720:  write using ('1,>') chr
721:  else
722:  if (chr >= 'a') and (chr <= 'f') then
723:  num = num*10h + (chr-57h)
724:  write using ('1,>') chr
725:  else
726:  if chr <> 0dh then write using('0,>') bell
727:  end
728:  end
729:  end
730:  end
731:  until chr == 0dh
732:  end
733:  return num
734:  end
735:
736:  define PROC getnum = DO
737:  define BYTE num
738:  define CHAR chr
```

280722-29

```
739:  define CHAR bell = 07h
740:  num = 0
741:  repeat
742:  chr = ci
743:  if (chr >= '0') and (chr <='9') then
744:  num = num*10t + (chr-30h)
745:  write using ('1,>') chr
746:  else
747:  if chr <> 0dh then write using('0,>') bell
748:  end
749:  end
750:  until chr == 0dh
751:  end
752:  return num
753:  end
754:
755:  define PROC getpl = DO
756:  curhome;cleareos
757:  define CHAR zzz
758:  write concat(1bh,'L',81h),'MAPIO COMMAND DISPLAYS OR SETS PHYSICAL LOCATION FOR I/O PORTS'
759:  write concat(1bh,'L',83h),' '
760:  write using('"                    HAVE YOU MAPPED THE I/O PORTS?(Y/N)",>') chrin
761:  write concat(1bh,'L',80h),' '
762:  repeat
763:  chrin = ci
764:  if not((chrin == 'N') or (chrin == 'n') or (chrin == 'Y') or (chrin == 'y')) then
765:  bell
766:  endif
767:  until (chrin == 'N') or (chrin == 'n') or (chrin == 'Y') or (chrin == 'y')
768:  end
769:  if (chrin == 'N') or (chrin == 'n') then do
770:  write using('0') chrin
771:  curhome;cleareos
772:  write '            THE I/O PORTS CAN BE MAPPED THE FOLLOWING WAY'
773:  write ' '
774:  write '                  MAPIO [(partition) <USER or ICE>]'
775:  write ''
776:  write '     MAPIO - displays the current map of I/O port address blocks'
777:  write ''
778:  write ' partition - is an entry specifying a range of addresses such as:'
779:  write '             Starting port-address LENGTH number of bytes'
780:  write ''
781:  write '      USER - Maps I/O to the USER system'
782:  write ''
783:  write '       ICE - Maps I/O to the I2ICE probe'
784:  write ''
785:  write concat(1bh,'L',83h),' PLEASE TYPE MAPIO STARTING HEXADECIMAL PORT-ADDRESS and <cr>'
786:  write concat(1bh,'L',80h), ' '
787:  paddr = gethex
788:  write 'n'
789:  write concat(1bh,'L',83h),'            PLEASE TYPE NUMBER OF BYTES and <cr>'
790:  write concat(1bh,'L',80h), ' '
791:  paddr1 = gethex
792:  write 'n'
793:  write concat(1bh,'L',83h),'          PLEASE TYPE ''U'' FOR USER or ''I'' FOR ICE'
794:  write concat(1bh,'L',80h),''
795:  repeat
796:  zzz = ci
797:  if not((zzz == 'U') or (zzz == 'u') or (zzz == 'I') or (zzz == 'i')) then
798:  bell
799:  endif
800:  until (zzz == 'U') or (zzz == 'u') or (zzz == 'I') or (zzz == 'i')
801:  end
802:  if (zzz=='I') or (zzz=='i') then
803:  MAPIO PADDR LENGTH PADDR1 ICE
804:  write using('"MAPIO ",0,n," LENGTH ",0,n," ICE "')paddr,paddr1
805:  write ''
806:  write concat(1bh,'L',83h),'                  PLEASE HIT ANY KEY TO CONTINUE'
807:  write concat(1bh,'L',80h),''
808:  zzz = ci;else
809:  if (zzz=='U') or (zzz=='u') then
810:  MAPIO PADDR LENGTH PADDR1 USER
811:  write using('"MAPIO ",0,n," LENGTH ",0,n," USER "')paddr,paddr1
812:  write ''
813:  write concat(1bh,'L',83h),'                  PLEASE HIT ANY KEY TO CONTINUE'
814:  write concat(1bh,'L',80h),''
815:  chrin = ci;endif;end
816:  end
817:  end
818:  curhome;cleareos
819:  write concat(1bh,'L',83h),'     TYPE IN THE HEXADECIMAL ADDRESS FOR THE 82530 and <cr>'
820:  write concat(1bh,'L',80h),' '
```

280722-30

```
821:  wreg = gethex
822:  write 'n'
823:  write ''
824:  end
825:
826:  define PROC getp2 = DO
827:  curhome;cleareos
828:  write concat(1bh,'L',81h),'                          ********************'
829:  write concat(1bh,'L',81h),'                          *PIN FUNCTION MENU**'
830:  write concat(1bh,'L',81h),'                          ********************'
831:  write concat(1bh,'L',80h),' '
832:  write ' '
833:  write ' '
834:  write '                      "R"  -  READ A REGISTER'
835:  write '                      "W"  -  WRITE TO A REGISTER'
836:  write '                      "N"  -  CHANGE 82530 PORT ADDRESS'
837:  write '                      "E"  -  EXIT TO MAIN MENU'
838:  write ' '
839:  write ' '
840:  write concat(1bh,'L',83h),'                          PLEASE TYPE YOUR CHOICE'
841:  write concat(1bh,'L',80h),' '
842:  end
843:
844:  define PROC getp3 = DO
845:  curhome;cleareos
846:  write ' '
847:  write concat(1bh,'L',81h),'      ******************************************** '
848:  write concat(1bh,'L',81h),'      THE CHOICES OF REGISTERS FOR YOUR SELECTION ARE :'
849:  write concat(1bh,'L',81h),'      ********************************************'
850:  write concat(1bh,'L',80h),' '
851:  if (chrin == 'R') or (chrin == 'r') then
852:  write '      0. RR0              1. RR1              2. RR2'
853:  write '      3. RR3              8. RR8             10. RR10'
854:  write '     12. RR12            13. RR13            15. RR15'
855:  else
856:  write '      0. WR0              1. WR1              2. WR2'
857:  write '      3. WR3              4. WR4              5. WR5'
858:  write '      6. WR6              7. WR7              8. WR8'
859:  write '      9. WR9             10. WR10            11. WR11'
860:  write '     12. WR12            13. WR13            14. WR14'
861:  write '                         15. WR15'
862:  end
863:  write ' '
864:  write ' '
865:  write concat(1bh,'L',83h),'               PLEASE TYPE YOUR CHOICE and <cr>'
866:  write concat(1bh,'L',80h),' '
867:  regnum = getnum
868:  write 'T'
869:  write ''
870:  end
871:
872:  define PROC getmen1 = DO
873:  if check then
874:  getp1
875:  check = false
876:  end
877:  repeat
878:  getp2
879:  chrin = ci
880:  if (chrin == 'E') or (chrin == 'e') then write using ('0') chrin;return end
881:  if (chrin == 'R') or (chrin == 'r') or (chrin == 'W') or (chrin == 'w') then
882:  write using ('0') chrin
883:  curhome;cleareos
884:  getp3
885:  return
886:  else
887:  if (chrin == 'N') or (chrin == 'n') then
888:  write using ('0') chrin
889:  getp1
890:  else
891:  if (chrin == 1bh) then return end
892:  bell
893:  end
894:  end
895:  end
896:  end
897:
898:  define PROC w_com1 = DO
899:  curhome;cleareos
900:  if %0 == 'e' then
901:  lst:job.inc
902:  write 'remove done,gethex,getnum,getmen1,w_com1,getp1,getp2,getp3'
```

280722-31

```
write "remove chrin,check,wreg,regnum,paddr,paddrl"
write "182530"
nolist
else
lst:job.inc
write using ("""incc:",0," nolist""") %0
write using ("0") %0
write using ("""remove ",0") %0
write 'incc:pinfum.inc nolist'
nolist
end
end

define PROC done = DO
curhome ; cleareos
getmenl
if (chrin == 'E') or (chrin == 'e') then
w_coml('e')
return
end
repeat
if (chrin == 'R') or (chrin == 'r') then
do
if regnum == 0 then
w_coml('rr0') ; return
end
if regnum == 1t then
w_coml('rr1');return
end
if regnum == 2t then
w_coml('rr2'); return
end
if regnum == 3t then
w_coml('rr3'); return
end
if regnum == 8t then
w_coml('rr8'); return
end
if regnum == 10t then
w_coml('rr10'); return
end
if regnum == 12t then
w_coml('rr12'); return
end
if regnum == 13t then
w_coml('rr13'); return
end
if regnum == 15t then
w_coml('rr15'); return
end
end
end
if (chrin == 'R') or (chrin == 'r')  then
write using ('"UNABLE TO READ REGISTER ",0,>') regnum
getp3
end
if (chrin == 'W') or (chrin == 'w') then
do
if regnum == 0 then
w_coml('wr0'); return
end
if regnum == 1t then
w_coml('wr1'); return
end
if regnum == 2t then
w_coml('wr2'); return
end
if regnum == 3t then
w_coml('wr3'); return
end
if regnum == 4t then
w_coml('wr4'); return
end
if regnum == 5t then
w_coml('wr5'); return
end
if regnum == 6t then
w_coml('wr6'); return
end
if regnum == 7t then
w_coml('wr7'); return
end
```

280722-32

```
 985:   if regnum == 8t then
 986:   w_coml('wr8'); return
 987:   end
 988:   if regnum == 9t then
 989:   w_coml('wr9'); return
 990:   end
 991:   if regnum == 10t then
 992:   w_coml('wr10'); return
 993:   end
 994:   if regnum == 11t then
 995:   w_coml('wr11'); return
 996:   end
 997:   if regnum == 12t then
 998:   w_coml('wr12'); return
 999:   end
1000:   if regnum == 13t then
1001:   w_coml('wr13'); return
1002:   end
1003:   if regnum == 14t then
1004:   w_coml('wr14'); return
1005:   end
1006:   if regnum == 15t then
1007:   w_coml('wr15'); return
1008:   end
1009:   end
1010:   end
1011:   if (chrin == 'W') or (chrin == 'w') then
1012:   write using ('*UNABLE TO WRITE TO REGISTER *,0,>') regnum
1013:   getp3
1014:   end
1015:   end
1016:   end
1017:
1018:
1019:
1020:                                   *PINFUN.INC*
1021:                                   ************
1022:
1023:   done
1024:   incc:job.inc nolist
1025:
1026:
1027:
1028:                                   *RR0*
1029:                                   *****
1030:
1031:   curhome;cleareos
1032:   write concat(1bh,'L',83h),'LOADING....'
1033:   write concat(1bh,'L',80h),''
1034:   define PROC rr0 = DO
1035:   define CHAR yyy
1036:   define BYTE temp
1037:   temp = port(wreg)
1038:   curhome ;cleareos
1039:   write concat(1bh,'L',81h),'                          READ REGISTER 0'
1040:   write concat(1bh,'L',81h),'                          ***************'
1041:   write concat(1bh,'L',80h),' '
1042:   write using ('2c,*=>*,2,y,5x,1,y,5x,1,y,5x,1,y,5x,1,y,5x,1,y,5x,1,y,5x,1,y') &
1043:   (temp and 80h) /80h, &
1044:   (temp and 40h) /40h, &
1045:   (temp and 20h) /20h, &
1046:   (temp and 10h) /10h, &
1047:   (temp and 08h) /08h, &
1048:   (temp and 04h) /04h, &
1049:   (temp and 02h) /02h, &
1050:   (temp and 01h) /01h
1051:   write concat(1bh,'L',80h),'_____________________________________________________'
1052:   write concat(1bh,'L',80h),'¶ D7  ¶ D6  ¶ D5  ¶ D4  ¶ D3  ¶ D2  ¶ D1  ¶ D0  ¶'
1053:   write concat(1bh,'L',80h),'-----------------------------------------------------'
1054:   write '    D7  =   BREAK/ABORT'
1055:   write '          D6  =   Tx UNDERRUN/EOM'
1056:   write '                D5  =   CTS'
1057:   write '                      D4  =   SYNC/nUNT'
1058:   write '                            D3  =   CD'
1059:   write '                                  D2  =   Tx BUFFER EMPTY'
1060:   write '                                        D1  =   ZERO COUNT'
1061:   write '                                              D0  =   Rx CmARACTER AVAILABLE'
1062:   write ' '
1063:   write concat(1Bh,'L',83h),'          PLEASE nIT ANY KEY TO RETURN       '
1064:   write concat(1bh,'L',80h), ' '
1065:   yyy=ci
1066:   end
```

280722-33

```
1067:
1068:
1069:
1070:                                   *RR1*
1071:                                   *****
1072:
1073:   curhome;cleareos
1074:   write concat(1bh,'L',83h),'LOADING....'
1075:   write concat(1bh,'L',80h),''
1076:   define PROC rr1 = DO
1077:   define CnAR yyy
1078:   define BYTE temp
1079:   temp = port(wreg)
1080:   curhome
1081:   cleareos
1082:   write concat(1bh,'L',81h),'                          READ REGISTER 1'
1083:   write concat(1bh,'L',81h),'                          ***************'
1084:   write concat(1bh,'L',80h),' '
1085:   write using ('2c,"=>",2,y,5x,1,y,5x,1,y,5x,1,y,5x,1,y,5x,1,y,5x,1,y,5x,1,y') &
1086:   (temp and 80h) /80h, &
1087:   (temp and 40h) /40h, &
1088:   (temp and 20h) /20h, &
1089:   (temp and 10h) /10h, &
1090:   (temp and 08h) /08h, &
1091:   (temp and 04h) /04h, &
1092:   (temp and 02h) /02h, &
1093:   (temp and 01h) /01h
1094:   write concat(1bh,'L',80h),'_____________________________________________________'
1095:   write concat(1bh,'L',80h),'¶ D7  ¶ D6  ¶ D5  ¶ D4  ¶ D3  ¶ D2  ¶ D1  ¶ D0  ¶'
1096:   write concat(1bh,'L',80h),'-----------------------------------------------------'
1097:   write '     D7  =   END OF FRAME'
1098:   write '           D6  =   CRC/FRAMING ERROR'
1099:   write '                 D5  =   Rx OVERRUN ERROR'
1100:   write '                       D4  =   PARITY ERROR'
1101:   write '                             D3  =   RESIDUE CODE 0'
1102:   write '                                   D2  =   RESIDUE CODE 1'
1103:   write '                                         D1  =   RESIDUE CODE 2'
1104:   write '                                               D0  =   ALL SENT'
1105:   write ' '
1106:   write concat(1bh,'L',83h),'          PLEASE HIT ANY KEY TO RETURN'
1107:   write concat(1bh,'l',80h), ' '
1108:   temp = ci
1109:   end
1110:
1111:
1112:
1113:                                   *RR2*
1114:                                   *****
1115:
1116:   curhome;cleareos
1117:   write concat(1bh,'L',83h),'LOADING....'
1118:   write concat(1bh,'L',80h),''
1119:   define PROC rr2 = DO
1120:   define CnAR yyy
1121:   define BYTE temp
1122:   temp = port(wreg)
1123:   curhome
1124:   cleareos
1125:   write concat(1bh,'L',81h),'                          READ REGISTER 2'
1126:   write concat(1bh,'L',81h),'                          ***************'
1127:   write concat(1bh,'L',80h),' '
1128:   write using ('2c,"=>",2,y,5x,1,y,5x,1,y,5x,1,y,5x,1,y,5x,1,y,5x,1,y,5x,1,y') &
1129:   (temp and 80h) /80h, &
1130:   (temp and 40h) /40h, &
1131:   (temp and 20h) /20h, &
1132:   (temp and 10h) /10h, &
1133:   (temp and 08h) /08h, &
1134:   (temp and 04h) /04h, &
1135:   (temp and 02h) /02h, &
1136:   (temp and 01h) /01h
1137:   write concat(1bh,'L',80h),'_____________________________________________________'
1138:   write concat(1bh,'L',80h),'¶ D7  ¶ D6  ¶ D5  ¶ D4  ¶ D3  ¶ D2  ¶ D1  ¶ D0  ¶'
1139:   write concat(1bh,'L',80h),'-----------------------------------------------------'
1140:   write '     D7  =   INTERRUPT VECTOR V7'
1141:   write '           D6  =   INTERRUPT VECTOR V6'
1142:   write '                 D5  =   INTERRUPT VECTOR V5'
1143:   write '                       D4  =   INTERRUPT VECTOR V4'
1144:   write '                             D3  =   INTERRUPT VECTOR V3'
1145:   write '                                   D2  =   INTERRUPT VECTOR V2'
1146:   write '                                         D1  =   INTERRUPT VECTOR V1'
1147:   write '                                               D0  =   INTERRUPT VECTOR V0'
1148:   write ' '
```

280722-34

```
1149:   write 'NOTE: Interrupt Vector is modified in B Channel'
1150:   write ' '
1151:   write concat(1Bh,'L',83h),'          PLEASE HIT ANY KEY TO RETURN'
1152:   write concat(1bh,'l',80h), ' '
1153:   temp=ci
1154:   end
1155:
1156:
1157:
1158:                                   *RR3*
1159:                                   *****
1160:
1161:   curhome;cleareos
1162:   write concat(1bh,'L',83h),'LOADING....'
1163:   write concat(1bh,'L',80h),''
1164:   define PROC rr3 = DO
1165:   define CHAR yyy
1166:   define BYTE temp
1167:   temp = port(wreg)
1168:   curhome
1169:   cleareos
1170:   write concat(1Bh,'L',81h),'                          READ REGISTER 3'
1171:   write concat(1bh,'L',81h),'                          ***************'
1172:   write concat(1bh,'L',80h),' '
1173:   write using ('2c,*=>*,2,y,5x,1,y,5x,1,y,5x,1,y,5x,1,y,5x,1,y,5x,1,y') &
1174:   (temp and 80h) /80h, &
1175:   (temp and 40h) /40h, &
1176:   (temp and 20h) /20h, &
1177:   (temp and 10h) /10h, &
1178:   (temp and 08h) /08h, &
1179:   (temp and 04h) /04h, &
1180:   (temp and 02h) /02h, &
1181:   (temp and 01h) /01h
1182:   write concat(1bh,'L',80h),'_____________________________________________'
1183:   write concat(1bh,'L',80h),'¶ D7 ¶ D6 ¶ D5 ¶ D4 ¶ D3 ¶ D2 ¶ D1 ¶ D0 ¶'
1184:   write concat(1bh,'L',80h),'---------------------------------------------'
1185:   write '     D7 =  0'
1186:   write '          D6 =  0'
1187:   write '               D5 =  CnANNEL A Rx IP'
1188:   write '                    D4 =  CHANNEL A Tx IP'
1189:   write '                         D3 =  CnANNEL A EXT/STAT IP'
1190:   write '                              D2 =  CHANNEL B Rx IP'
1191:   write '                                   D1 =  CnANNEL B Tx IP'
1192:   write '                                        D0 =  CHANNEL B EXT/STAT IP'
1193:   write ' '
1194:   write 'NOTE: Always in B Channel'
1195:   write ' '
1196:   write concat(1BH,'L',83H),'          PLEASE HIT ANY KEY TO RETURN'
1197:   write concat(1bh,'l',80h), ' '
1198:   temp=ci
1199:   end
1200:
1201:
1202:
1203:                                   *RR8*
1204:                                   *****
1205:
1206:   curhome;cleareos
1207:   write concat(1bh,'L',83h),'LOADING....'
1208:   write concat(1bh,'L',80h),''
1209:   define PROC rr3 = DO
1210:   define CnAR yyy
1211:   define BYTE temp
1212:   temp = port(wreg)
1213:   curhome
1214:   cleareos
1215:   write concat(1bh,'L',81h),'                          READ REGISTER 8'
1216:   write concat(1bh,'L',81h),'                          ***************'
1217:   write concat(1bh,'L',80h),' '
1218:   write using ('2c,*=>*,2,y,5x,1,y,5x,1,y,5x,1,y,5x,1,y,5x,1,y,5x,1,y,5x,1,y') &
1219:   (temp and 80h) /80h, &
1220:   (temp and 40h) /40h, &
1221:   (temp and 20h) /20h, &
1222:   (temp and 10h) /10h, &
1223:   (temp and 08h) /08h, &
1224:   (temp and 04h) /04h, &
1225:   (temp and 02h) /02h, &
1226:   (temp and 01h) /01h
1227:   write concat(1bh,'L',80h),'_____________________________________________'
1228:   write concat(1bh,'L',80h),'¶ D7 ¶ D6 ¶ D5 ¶ D4 ¶ D3 ¶ D2 ¶ D1 ¶ D0 ¶'
1229:   write concat(1bh,'L',80h),'---------------------------------------------'
1230:   write ''
```

280722-35

```
write ''
write ''
write ''
write ''
write ''
write ''
write ''
write ' '
write ''
write ' '
write concat(1Bn,'L',83n),'          PLEASE nIT ANY KEY TO RETURN'
write concat(1bh,'l',80h), ' '
temp=ci
end



                                  *RR10*
                                  ******

curhome;cleareos
write concat(1bh,'L',83h),'LOADING....'
write concat(1bh,'L',80h),''
define PROC rr10= DO
define CnAR yyy
define BYTE temp
temp  = port(wreg)
curhome
cleareos
write concat(1bh,'L',81h),'                        READ REGISTER 10'
write concat(1bh,'L',81h),'                        ****************'
write concat(1bh,'L',80h),' '
write using ('2c,"=>",2,y,5x,1,y,5x,1,y,5x,1,y,5x,1,y,5x,1,y,5x,1,y,5x,1,y') &
(temp and 80h) /80h, &
(temp and 40h) /40h, &
(temp and 20h) /20h, &
(temp and 10h) /10h, &
(temp and 08h) /08h, &
(temp and 04h) /04h, &
(temp and 02h) /02h, &
(temp and 01h) /01h
write concat(1bh,'L',80h),'_____________________________________________________'
write concat(1bh,'L',80h),'¶ D7  ¶ D6  ¶ D5  ¶ D4  ¶ D3  ¶ D2  ¶ D1  ¶ D0  ¶'
write concat(1bh,'L',80h),'-----------------------------------------------------'
write '    D7  =   ONE CLOCK MISSING'
write '          D6  =   TWO CLOCKS MISSING'
write '                D5  =   0'
write '                      D4  =   LOOP SENDING'
write '                            D3  =   0'
write '                                  D2  =   0'
write '                                        D1  =   ON LOOP'
write '                                              D0  =   0'
write ' '
write concat(1ba,'L',83h),'          PLEASE nIT ANY KEY TO RETURN'
write concat(1bh,'L',80h), ' '
temp=ci
end



                                  *RR12*
                                  ******

curhome;cleareos
write concat(1bh,'L',83h),'LOADING....'
write concat(1bh,'L',80h),''
define PROC rr12= DO
define CnAR yyy
define BYTE temp
temp  = port(wreg)
curhome
cleareos
write concat(1bh,'L',81h),'                        READ REGISTER 12'
write concat(1bh,'L',81h),'                        ****************'
write concat(1bh,'L',80h),' '
write using ('2c,"=>",2,y,5x,1,y,5x,1,y,5x,1,y,5x,1,y,5x,1,y,5x,1,y,5x,1,y') &
(temp and 80h) /80h, &
(temp and 40h) /40h, &
(temp and 20h) /20h, &
(temp and 10h) /10h, &
(temp and 08h) /08h, &
(temp and 04h) /04h, &
(temp and 02h) /02h, &
```

280722-36

```
(temp and 01h) /01h
write concat(1bh,'L',80h),'_____________________________________________'
write concat(1bh,'L',80h),'¶ D7 ¶ D6 ¶ D5 ¶ D4 ¶ D3 ¶ D2 ¶ D1 ¶ D0 ¶'
write concat(1bh,'L',80h),'---------------------------------------------'
write '    D7  =   TC7'
write '          D6  =   TC6'
write '                D5  =   TC5'
write '                      D4  =   TC4'
write '                            D3  =   TC3'
write '                                  D2  =   TC2'
write '                                        D1  =   TC1'
write '                                              D0  =   TC0'
write ' '
write 'NOTE: Lower Byte of TIME CONSTANT'
write ' '
write concat(1Bh,'L',83h),'          PLEASE HIT ANY KEY TO RETURN'
write concat(1bh,'l',80h), ' '
temp=ci
end



                              *RR13*
                              ******

curhome;cleareos
write concat(1bh,'L',83h),'LOADING....'
write concat(1bh,'L',80h),''
define PROC rr13= DO
define CHAR yyy
define BYTE temp
temp  = port(wreg)
curhome
cleareos
write concat(1bh,'L',81h),'                    READ REGISTER 13'
write concat(1bh,'L',81h),'                    ****************'
write concat(1bh,'L',80h),' '
write using ('2c,*=>*,2,y,5x,1,y,5x,1,y,5x,1,y,5x,1,y,5x,1,y,5x,1,y') &
(temp and 80h) /80h, &
(temp and 40h) /40h, &
(temp and 20h) /20h, &
(temp and 10h) /10h, &
(temp and 08h) /08h, &
(temp and 04h) /04h, &
(temp and 02h) /02h, &
(temp and 01h) /01h
write concat(1bh,'L',80h),'_____________________________________________'
write concat(1bh,'L',80h),'¶ D7 ¶ D6 ¶ D5 ¶ D4 ¶ D3 ¶ D2 ¶ D1 ¶ D0 ¶'
write concat(1bh,'L',80h),'---------------------------------------------'
write '    D7  =   TC15'
write '          D6  =   TC14'
write '                D5  =   TC13'
write '                      D4  =   TC12'
write '                            D3  =   TC11'
write '                                  D2  =   TC10'
write '                                        D1  =   TC9'
write '                                              D0  =   TC8'
write ' '
write 'NOTE: Upper Byte of TIME CONSTANT'
write ' '
write concat(1Bh,'L',83H),'          PLEASE HIT ANY KEY TO RETURN'
write concat(1bh,'l',80h), ' '
temp=ci
end



                              *RR15*
                              ******

curhome;cleareos
write concat(1bh,'L',83h),'LOADING....'
write concat(1bh,'L',80h),''
define PROC rr15= DO
define CHAR yyy
define BYTE temp
temp = port(wreg)
curhome
cleareos
write concat(1bh,'L',81h),'                    READ REGISTER 15'
write concat(1bh,'L',81h),'                    ****************'
write concat(1bh,'L',80h),' '
```

280722-37

```
1396:   write using ('2c,"=>",2,y,5x,1,y,5x,1,y,5x,1,y,5x,1,y,5x,1,y,5x,1,y,5x,1,y,5x,1,y,5x,1,y') &
1397:   (temp and 80h) /80h, &
1398:   (temp and 40h) /40h, &
1399:   (temp and 20h) /20h, &
1400:   (temp and 10h) /10h, &
1401:   (temp and 08h) /08h, &
1402:   (temp and 04h) /04h, &
1403:   (temp and 02h) /02h, &
1404:   (temp and 01h) /01h
1405:   write concat(1bh,'L',80h),'_____________________________________________'
1406:   write concat(1bh,'L',80h),'¶ D7 ¶ D6 ¶ D5 ¶ D4 ¶ D3 ¶ D2 ¶ D1 ¶ D0 ¶'
1407:   write concat(1bh,'L',80h),'---------------------------------------------'
1408:   write '    D7  =  BREAK/ABORT IE'
1409:   write '         D6  =  Tx UNDERRUN/EOM IE'
1410:   write '              D5  =  CTS IE'
1411:   write '                   D4  =  SYNC/nUNT IE'
1412:   write '                        D3  =  CD IE'
1413:   write '                             D2  =  0'
1414:   write '                                  D1  =  ZERO COUNT IE'
1415:   write '                                       D0  =  0'
1416:   write ' '
1417:   write concat(1BH,'L',83H),'          PLEASE HIT ANY KEY TO RETURN'
1418:   write concat(1bh,'l',80h), ' '
1419:   temp=ci
1420:   end
1421:
1422:
1423:
1424:                                *WR0*
1425:                                *****
1426:
1427:   curhome;cleareos
1428:   write concat(1bh,'L',83h),'LOADING....'
1429:   write concat(1bh,'L',80h),''
1430:  define PROC wr0 = DO
1431:   define CHAR yyy
1432:   define BYTE temp
1433:   curhome;cleareos
1434:   write concat(1bh,'L',81h),'                    WRITE REGISTER 0'
1435:   write concat(1bh,'L',81h),'                    ****************'
1436:   write concat(1bh,'L',80h),'_____________________________________________'
1437:   write concat(1bh,'L',80h),'¶ D7 ¶ D6 ¶ D5 ¶ D4 ¶ D3 ¶ D2 ¶ D1 ¶ D0 ¶'
1438:   write concat(1bh,'L',80h),'---------------------------------------------'
1439:   write '                        D7 D6'
1440:   write '                        0  0  =  NULL CODE '
1441:   write '                        0  1  =  RESET Rx CRC CHECKER'
1442:   write '                        1  0  =  RESET Tx CRC GENERATOR'
1443:   write '                        1  1  =  RESET Tx UNDERRUN/EOM LATCH'
1444:   write ' D5 D4 D3                                    D2 D1 D0'
1445:   write ' 0  0  0  =  NULL CODE                       0  0  0  =  0 or  8'
1446:   write ' 0  0  1  =  POINT nIGn REGISTER GROUP       0  0  1  =  1 or  9'
1447:   write ' 0  1  0  =  PRESEI EXT/STAIUS INTERRUPTS    0  1  0  =  2 or 10'
1448:   write ' 0  1  1  =  SEND ABORT                      0  1  1  =  3 or 11'
1449:   write ' 1  0  0  =  ENABLE INT ON NEXT Rx CnAR.     1  0  0  =  4 or 12'
1450:   write ' 1  0  1  =  RESET TxINI PENDING             1  0  1  =  5 or 13'
1451:   write ' 1  1  0  =  ERROR RESET                     1  1  0  =  6 or 14'
1452:   write ' 1  1  1  =  RESET HIGHEST IUS              1  1  1  =  7 or 15'
1453:   write concat(1bh,'L',83h),'   PLEASE TYPE THE VALUE TO BE WRITTEN and <cr>'
1454:   write concat(1bh,'L',80h),' '
1455:   temp = gethex
1456:   write 'n'
1457:   port(wreg)=0
1458:   port(wreg) = temp
1459:   write ' '
1460:   write concat(1Bn,'L',83n),'          PLEASE nIT ANY KEY TO RETURN'
1461:   write concat(1bh,'l',80h), ' '
1462:   yyy=ci
1463:   end
1464:
1465:
1466:
1467:                                *WR1*
1468:                                *****
1469:
1470:   curhome;cleareos
1471:   write concat(1bh,'L',83h),'LOADING....'
1472:   write concat(1bh,'L',80h),''
1473:  define PROC wr1 = DO
1474:   define CnAR yyy
1475:   define BYTE temp
1476:   curhome;cleareos
```

280722-38

```
1477:  write concat(1bh,'L',81h),'                    WRITE REGISTER 1'
1478:  write concat(1bh,'L',81h),'                    ****************'
1479:  write concat(1bh,'L',80h),'-------------------------------------------------'
1480:  write concat(1bh,'L',80h),'¶ D7  ¶ D6  ¶ D5  ¶ D4  ¶ D3  ¶ D2  ¶ D1  ¶ D0  ¶'
1481:  write concat(1bh,'L',80h),'-------------------------------------------------'
1482:  write '    D7  =   READY/DMA REQUEST ENABLE'
1483:  write '          D6  =   READY/DMA REQUEST FUNCTION'
1484:  write '                D5  =   READY/DMA REQUEST ON RECEIVE/TRANSMIT'
1485:  write '                      D4    D3   =  0 0  =  Rx INT DISABLE'
1486:  write '                                 =  0 1  =  Rx INT ON FIRST CnAR OR CONDN'
1487:  write '                                 =  1 0  =  INT ON ALL Rx CHAR OR CONDN'
1488:  write '                                 =  1 1  =  Rx INT ON SPECIAL CONDN ONLY'
1489:  write '                                  D2  =   PARITY IS SPECIAL CONDITION'
1490:  write '                                      D1  =   Tx INT ENABLE'
1491:  write '                                          D0  =   EXT. INT ENABLE'
1492:  write concat(1bh,'L',83h),'   PLEASE TYPE THE VALUE TO BE WRITTEN and <cr>'
1493:  write concat(1bh,'L',80h),' '
1494:  temp = gethex
1495:  write 'H'
1496:  port(wreg) = 1t
1497:  port(wreg) = temp
1498:  write concat(1BH,'L',83H),'          PLEASE HIT ANY KEY TO RETURN'
1499:  write concat(1bh,'l',80h),' '
1500:  yyy = ci
1501:  end
1502:
1503:
1504:
1505:                                  *WR2*
1506:                                  *****
1507:
1508:  curhome;cleareos
1509:  write concat(1bh,'L',83h),'LOADING....'
1510:  write concat(1bh,'L',80h),''
1511:  define PROC wr2 = DO
1512:  define CHAR yyy
1513:  define BYTE temp
1514:  curhome;cleareos
1515:  write concat(1bh,'L',81h),'                    WRITE REGISTER 2'
1516:  write concat(1bh,'L',81h),'                    ****************'
1517:  write concat(1bh,'L',80h),'-------------------------------------------------'
1518:  write concat(1bh,'L',80h),'¶ D7  ¶ D6  ¶ D5  ¶ D4  ¶ D3  ¶ D2  ¶ D1  ¶ D0  ¶'
1519:  write concat(1bh,'L',80h),'-------------------------------------------------'
1520:  write '    D7  =   INTERRUPT VECTOR V7'
1521:  write '          D6  =   INTERRUPT VECTOR V6'
1522:  write '                D5  =   INTERRUPT VECTOR V5'
1523:  write '                      D4  =   INTERRUPT VECTOR V4'
1524:  write '                            D3  =   INTERRUPT VECTOR V3'
1525:  write '                                  D2  =   INTERRUPT VECTOR V2'
1526:  write '                                        D1  =   INTERRUPT VECTOR V1'
1527:  write '                                            D0  =   INTERRUPT VECTOR V0'
1528:  write concat(1bh,'L',83h),'   PLEASE TYPE THE VALUE TO BE WRITTEN and <cr>'
1529:  write concat(1bh,'L',80h),' '
1530:  temp = gethex
1531:  write 'n'
1532:  port(wreg) = 2t
1533:  port(wreg) = temp
1534:  write concat(1Bn,'L',83n),'          PLEASE HIT ANY KEY TO RETURN'
1535:  write concat(1bh,'l',80h),' '
1536:  yyy=ci
1537:  end
1538:
1539:
1540:
1541:                                  *WR3*
1542:                                  *****
1543:
1544:  curhome;cleareos
1545:  write concat(1bh,'L',83h),'LOADING....'
1546:  write concat(1bh,'L',80h),''
1547:  define PROC wr3 = DO
1548:  define CnAR yyy
1549:  define BYTE temp
1550:  curhome ;cleareos
1551:  write concat(1bh,'L',81h),'                    WRITE REGISTER 3'
1552:  write concat(1bh,'L',81h),'                    ****************'
1553:  write concat(1bh,'L',80h),'-------------------------------------------------'
1554:  write concat(1bh,'L',80h),'¶ D7  ¶ D6  ¶ D5  ¶ D4  ¶ D3  ¶ D2  ¶ D1  ¶ D0  ¶'
1555:  write concat(1bh,'L',80h),'-------------------------------------------------'
1556:  write '    D7      D6  =  0 0  =  Rx 5 BITS/CnARACTER'
1557:  write '                =  0 1  =  Rx 7 BITS CHARACTER'
1558:  write '                =  1 0  =  Rx 6 BITS CnARACTER'
```

280722-39

```
1559:  write '                = 1 1  =  Rx 8 BITS CHARACTER'
1560:  write '                 D5  =   AUTO ENABLES'
1561:  write '                       D4  =   ENTER HUNT MODE'
1562:  write '                             D3  =   Rx CRC ENABLE'
1563:  write '                                   D2  =   ADDRESS SEARCH MODE(SDLC)'
1564:  write '                                         D1  =   SYNC CHAR. LOAD INHIBIT'
1565:  write '                                               D0  =   Rx ENABLE'
1566:  write concat(1bh,'L',83h),'   PLEASE TYPE THE VALUE TO BE WRITTEN and <cr>'
1567:  write concat(1bh,'L',80h),' '
1568:  temp = gethex
1569:  write 'n'
1570:  port(wreg) = 3t
1571:  port(wreg) = temp
1572:  write concat(1Bn,'L',83n),'           PLEASE nIT ANY KEY TO RETURN'
1573:  write concat(1bh,'1',80h),' '
1574:  yyy=ci
1575:  end
1576:
1577:
1578:
1579:                                  *WR4*
1580:                                  *****
1581:
1582:  curhome;cleareos
1583:  write concat(1bh,'L',83h),'LOADING....'
1584:  write concat(1bh,'L',80h),''
1585:  define PROC wr4 = DO
1586:  define CHAR yyy
1587:  define BYTE temp
1588:  curhome ;cleareos
1589:  write concat(1bh,'L',81h),'                          WRITE REGISTER 4'
1590:  write concat(1bh,'L',81h),'                          ****************'
1591:  write concat(1bh,'L',80h),'_____________________________________________'
1592:  write concat(1bh,'L',80h),'¶ D7  ¶ D6  ¶ D5  ¶ D4  ¶ D3  ¶ D2  ¶ D1  ¶ D0  ¶'
1593:  write concat(1bh,'L',80h),'---------------------------------------------'
1594:  write '    D7     D6  =  0 0  =  X1 CLOCK MODE'
1595:  write '               =  0 1  =  X16 CLOCK MODE'
1596:  write '               =  1 0  =  X32 CLOCK MODE'
1597:  write '               =  1 1  =  X64 CLOCK MODE'
1598:  write '                D5     D4  =  0 0  =  8 BIT SYNC CHARACTER'
1599:  write '                           =  0 1  =  16 BIT SYNC CaARACTER'
1600:  write '                           =  1 0  =  SDLC MODE(01111110 FLAG)'
1601:  write '                           =  1 1  =  EXTERNAL SYNC MODE '
1602:  write '                            D3      D2  =  0 0  =  SYNC MODES ENABLE'
1603:  write '                                       =  0 1  =  1 STOP BIT/CaARACTER'
1604:  write '                                       =  1 0  =  1.5 STOP BITS/CaAR.'
1605:  write '                                       =  1 1  =  2 STOP BITS/CHAR.'
1606:  write '                                       D1  =   PARITY EVEN/ODD'
1607:  write '                                             D0  =   PARITY ENABLE'
1608:  write 'NOTE: Interrupt Vector is modified in B Channel'
1609:  write concat(1bh,'L',83h),'   PLEASE TYPE THE VALUE TO BE WRITTEN and <cr>'
1610:  write concat(1bh,'L',80h),' '
1611:  temp = gethex
1612:  write 'H'
1613:  port(wreg) = 4t
1614:  port(wreg) = temp
1615:  write concat(1Bn,'L',83n),'           PLEASE nIT ANY KEY TO RETURN'
1616:  write concat(1bh,'1',80h),' '
1617:  yyy=ci
1618:  end
1619:
1620:
1621:
1622:                                  *WR5*
1623:                                  *****
1624:
1625:  curhome;cleareos
1626:  write concat(1bh,'L',83h),'LOADING....'
1627:  write concat(1bh,'L',80h),''
1628:  define PROC wr5= DO
1629:  define CnAR yyy
1630:  define BYTE temp
1631:  curhome ;cleareos
1632:  write concat(1bh,'L',81h),'                          WRITE REGISTER 5'
1633:  write concat(1bh,'L',81h),'                          ****************'
1634:  write concat(1bh,'L',80h),'_____________________________________________'
1635:  write concat(1bh,'L',80h),'¶ D7  ¶ D6  ¶ D5  ¶ D4  ¶ D3  ¶ D2  ¶ D1  ¶ D0  ¶'
1636:  write concat(1bh,'L',80h),'---------------------------------------------'
1637:  write '    D7  =   DTR'
1638:  write '          D6  =   TWO CLOCKS MISSING'
1639:  write '               D5     D4  =  0 0  =  Tx 5 BITS(OR LESS)/CaARACTER'
1640:  write '                          =  0 1  =  Tx 7 BITS/CHARACTER'
```

280722-40

```
1641:   write '                                  =  1 0  =  Tx 6 BITS/CHARACTER'
1642:   write '                                  =  1 1  =  Tx 8 BITS/CHARACTER'
1643:   write '                                 D3  =   Tx ENABLE'
1644:   write '                                        D2  =   (NOT)SDLC/CRC-16'
1645:   write '                                              D1  =   RTS'
1646:   write '                                                    D0  =   Tx CRC ENABLE'
1647:   write concat(1bh,'L',83h),'   PLEASE TYPE THE VALUE TO BE WRITTEN and <cr>'
1648:   write concat(1bh,'L',80h),' '
1649:   temp = gethex
1650:   write '|'
1651:   port(wreg) = 5t
1652:   port (wreg) = temp
1653:   write concat(1BH,'L',83H),'          PLEASE HIT ANY KEY TO RETURN'
1654:   write concat(1bh,'1',80h),  ' '
1655:   yyy=ci
1656:   end
1657:
1658:
1659:
1660:                                  *WR6*
1661:                                  *****
1662:
1663:   curhome;cleareos
1664:   write concat(1bh,'L',83h),'LOADING....'
1665:   write concat(1bh,'L',80h),''
1666:   define PROC wr6 = DO
1667:   define CHAR yyy
1668:   define BYTE temp
1669:   curhome ;cleareos
1670:   write concat(1bh,'L',81h),'                          WRITE REGISTER 6'
1671:   write concat(1bh,'L',81h),'                          ****************'
1672:   write concat(1bh,'L',80h),'_____________________________________________'
1673:   write concat(1bh,'L',80h),'¶ D7  ¶ D6  ¶ D5  ¶ D4  ¶ D3  ¶ D2  ¶ D1  ¶ D0  ¶'
1674:   write concat(1bh,'L',80h),'---------------------------------------------'
1675:   write '   SYNC7 SYNC6 SYNC5 SYNC4 SYNC3 SYNC2 SYNC1 SYNC0      MONOSYNC 8 BITS'
1676:   write '   SYNC1 SYNC0 SYNC5 SYNC4 SYNC3 SYNC2 SYNC1 SYNC0      MONOSYNC 8 BITS'
1677:   write ' '
1678:   write '   SYNC7 SYNC6 SYNC5 SYNC4 SYNC3 SYNC2 SYNC1 SYNC0      BISYNC 16 BITS'
1679:   write '   SYNC3 SYNC2 SYNC1 SYNC0   1     1     1     1       BISYNC 12 BITS'
1680:   write ' '
1681:   write '   ADR7  ADR6  ADR5  ADR4  ADR3  ADR2  ADR1  ADR0      SDLC'
1682:   write '   ADR7  ADR6  ADR5  ADR4    1     1     1     1       SDLC(ADDRESS 0)'
1683:   write ''
1684:   write concat(1bh,'L',83h),'   PLEASE TYPE THE VALUE TO BE WRITTEN and <cr>'
1685:   write concat(1bh,'L',80h),' '
1686:   temp = gethex
1687:   write 'n'
1688:   port(wreg) = 6t
1689:   port(wreg) = temp
1690:   write concat(1Bn,'L',83n),'          PLEASE nIT ANY KEY TO RETURN'
1691:   write concat(1bh,'1',80h),  ' '
1692:   yyy=ci
1693:   end
1694:
1695:
1696:
1697:                                  *WR7*
1698:                                  *****
1699:
1700:   curhome;cleareos
1701:   write concat(1bh,'L',83h),'LOADING....'
1702:   write concat(1bh,'L',80h),''
1703:   define PROC wr7 = DO
1704:   define CnAR yyy
1705:   define BYTE temp
1706:   curhome ;cleareos
1707:   write concat(1bh,'L',81h),'                          WRITE REGISTER 7'
1708:   write concat(1bh,'L',81h),'                          ****************'
1709:   write concat(1bh,'L',80h),'_____________________________________________'
1710:   write concat(1bh,'L',80h),'¶ D7  ¶ D6  ¶ D5  ¶ D4  ¶ D3  ¶ D2  ¶ D1  ¶ D0  ¶'
1711:   write concat(1bh,'L',80h),'---------------------------------------------'
1712:   write '  SYNC7  SYNC6  SYNC5  SYNC4  SYNC3  SYNC2  SYNC1 SYNC0      MONOSYNC 8 BITS'
1713:   write '  SYNC5  SYNC4  SYNC3  SYNC2  SYNC1  SYNC0    1     1        MONOSYNC 8 BITS'
1714:   write ' '
1715:   write '  SYNC5  SYNC14 SYNC13 SYNC12 SYNC11 SYNC10 SYNC9 SYNC8     BISYNC 16 BITS'
1716:   write '  SYNC11 SYNC10 SYNC9  SYNC8  SYNC7  SYNC6  SYNC5 SYNC4     BISYNC 12 BITS'
1717:   write ' '
1718:   write '     0      1      1      1      1      1      1      0        SDLC'
1719:   write ''
1720:   write concat(1bh,'L',83h),'   PLEASE YTPE THE VALUE TO BE WRITTEN and <cr>'
1721:   write concat(1bh,'L',80h),' '
1722:   temp = gethex
```

280722-41

```
1723:   write 'n'
1724:   port(wreg) = 7t
1725:   port(wreg) = temp
1726:   write concat(1Bn,'L',83n),'          PLEASE nIT ANY KEY TO RETURN'
1727:   write concat(1bh,'1',80h), ' '
1728:   yyy=ci
1729:   end
1730:
1731:
1732:
1733:                                   *WR8*
1734:                                   *****
1735:
1736:   curhome;cleareos
1737:   write concat(1bh,'L',83h),'LOADING....'
1738:   write concat(1bh,'L',80h),''
1739:   define PROC wr8 = DO
1740:   define CnAR yyy
1741:   define BYTE temp
1742:   curhome;cleareos
1743:   write concat(1bh,'L',81h),'                          WRITE REGISTER 8'
1744:   write concat(1bh,'L',81h),'                          ****************'
1745:   write concat(1bh,'L',80h),'_____________________________________________'
1746:   write concat(1bh,'L',80h),'¶ D7  ¶ D6  ¶ D5  ¶ D4  ¶ D3  ¶ D2  ¶ D1  ¶ D0  ¶'
1747:   write concat(1bh,'L',80h),'---------------------------------------------'
1748:   write ' '
1749:   write ' '
1750:   write ' '
1751:   write ' '
1752:   write ' '
1753:   write ' '
1754:   write ' '
1755:   write ' '
1756:   write concat(1bh,'L',83h),'   PLEASE TYPE TnE VALUE TO BE WRITTEN and <cr>'
1757:   write concat(1bh,'L',80h),' '
1758:   temp = gethex
1759:   write 'H'
1760:   port(wreg) = 2t
1761:   port(wreg) = temp
1762:   write concat(1Bn,'L',83n),'          PLEASE nIT ANY KEY TO RETURN'
1763:   write concat(1bh,'1',80h), ' '
1764:   yyy=ci
1765:   end
1766:
1767:
1768:
1769:                                   *WR9*
1770:                                   *****
1771:
1772:   curhome;cleareos
1773:   write concat(1bh,'L',83h),'LOADING....'
1774:   write concat(1bh,'L',80h),''
1775:   define PROC wr9= DO
1776:   define CnAR yyy
1777:   define BYTE temp
1778:   curhome ;cleareos
1779:   write concat(1bh,'L',81h),'                          WRITE REGISTER 9'
1780:   write concat(1bh,'L',81h),'                          ****************'
1781:   write concat(1bh,'L',80h),'_____________________________________________'
1782:   write concat(1bh,'L',80h),'¶ D7  ¶ D6  ¶ D5  ¶ D4  ¶ D3  ¶ D2  ¶ D1  ¶ D0  ¶'
1783:   write concat(1bh,'L',80h),'---------------------------------------------'
1784:   write '    D7     D6  =  0 0  =  NO RESET'
1785:   write '                =  0 1  =  CHANNEL RESET B'
1786:   write '                =  1 0  =  CnANNEL RESET A'
1787:   write '                =  1 1  =  FORCE HARDWARE RESET'
1788:   write '                 D5  =   0'
1789:   write '                       D4  =   STATUS HIGH/(NOT)STATUS LOW'
1790:   write '                             D3  =   MIE'
1791:   write '                                   D2  =   DLC'
1792:   write '                                         D1  =   NV'
1793:   write '                                               D0  =   VIS'
1794:   write concat(1bh,'L',83h),'   PLEASE TYPE THE VALUE TO BE WRITTEN and <cr>'
1795:   write concat(1bh,'L',80h),' '
1796:   temp = gethex
1797:   write 'n'
1798:   port(wreg) = 9t
1799:   port(wreg) = temp
1800:   write concat(1Bn,'L',83n),'          PLEASE nIT ANY KEY TO RETURN'
1801:   write concat(1bh,'1',80h), ' '
1802:   yyy=ci
1803:   end
1804:
```

280722-42

```
1805:
1806:
1807:                                   *WR10*
1808:                                   ******
1809:
1810:  curhome;cleareos
1811:  write concat(1bh,'L',83h),'LOADING....'
1812:  write concat(1bh,'L',80h),''
1813:  define PROC wr10 = DO
1814:  define CaAR yyy
1815:  define BYTE temp
1816:  curhome ;cleareos
1817:  write concat(1bh,'L',81h),'                          WRITE REGISTER 10'
1818:  write concat(1bh,'L',81h),'                          ****************'
1819:  write concat(1bh,'L',80h),'_____________________________________________'
1820:  write concat(1bh,'L',80h),'¶ D7  ¶ D6  ¶ D5  ¶ D4  ¶ D3  ¶ D2  ¶ D1  ¶ D0  ¶'
1821:  write concat(1bh,'L',80h),'---------------------------------------------'
1822:  write '    D7  =   CRC PRESET I/(NOT)O'
1823:  write '          D6     D5  =  0 0  =  NRZ'
1824:  write '                        =  0 1  =  NRZI'
1825:  write '                        =  1 0  =  FM1 (TRANSMISSION 1)'
1826:  write '                        =  1 1  =  FM0 (TRANSMISSION 0)'
1827:  write '                         D4  =   GO ACTIVE ON ROLL'
1828:  write '                               D3  =   MARK/(NOT)FLAG IDEL'
1829:  write '                                     D2  =   ABORT/(NOT)FLAG ON UNDERRRUN'
1830:  write '                                           D1  =   LOOP MODE  _________'
1831:  write '                                                 D0  =   6 BIT/8 BIT SYNC'
1832:  write concat(1bh,'L',83h),'   PLEASE TYPE THE VALUE TO BE WRITTEN and <cr>'
1833:  write concat(1bh,'L',80h),' '
1834:  temp = gethex
1835:  write 'H'
1836:  port(wreg) = 10t
1837:  port(wreg) = temp
1838:  write concat(1Bh,'L',83h),'          PLEASE HIT ANY KEY TO RETURN'
1839:  write concat(1bh,'l',80h),' '
1840:  yyy=ci
1841:  end
1842:
1843:
1844:
1845:                                   *WR11*
1846:                                   ******
1847:
1848:  curhome;cleareos
1849:  write concat(1bh,'L',83h),'LOADING....'
1850:  write concat(1bh,'L',80h),''
1851:  define PROC wr11 = DO
1852:  define CaAR yyy
1853:  define BYTE temp
1854:  curhome ;cleareos
1855:  write concat(1bh,'L',81h),'                          WRITE REGISTER 11'
1856:  write concat(1bh,'L',81h),'                          ****************'
1857:  write concat(1bh,'L',80h),'_____________________________________________'
1858:  write concat(1bh,'L',80h),'¶ D7  ¶ D6  ¶ D5  ¶ D4  ¶ D3  ¶ D2  ¶ D1  ¶ D0  ¶'
1859:  write concat(1bh,'L',80h),'---------------------------------------------'
1860:  write '    D7  =   (NOT)RTxC XTAL/(NOT)NO XTAL  ____'
1861:  write '          D6     D5  =  0 0   =  RECEIVE CLOCK = RTxC PIN'
1862:  write '                        =  0 1   =  RECEIVE CLOCK = (NOT)TRxC PIN'
1863:  write '                        =  1 0   =  RECEIVE CLOCK = BR GENERATOR OUTPUT'
1864:  write '                        =  1 1   =  RECEIVE CLOCK = DPLL OUTPUT'
1865:  write '                         D4     D3  =  0 0  =  TRANSMIT CLOCK = (NOT)RTxC PIN'
1866:  write '                                    =  0 1  =  TRANSMIT CLOCK = (NOT)TRxC PIN'
1867:  write '                                    =  1 0  =  TRANSMIT CLOCK = BR GEN. OUTPUT'
1868:  write '                                    =  1 1  =  TRANSMIT CLOCK = DPLL OUTPUT'
1869:  write '                                     D2  =   (NOT)TRxC O/1'
1870:  write '                                           D1     D0 = 0 0 = XTAL OUTPUT'
1871:  write '                                                       = 0 1 = TRANSMIT CLOCK'
1872:  write '                                                       = 1 0 = BR GENERATOR O/P'
1873:  write '                                                       = 1 1 = DPLL OUTPUT'
1874:  write concat(1bh,'L',83h),'   PLEASE YTPE THE VALUE TO BE WRITTEN and <cr>'
1875:  write concat(1bh,'L',80h),' '
1876:  temp = gethex
1877:  write 'H'
1878:  port(wreg) = 11t
1879:  port(wreg) = temp
1880:  write concat(1Bh,'L',83h),'          PLEASE HIT ANY KEY TO RETURN'
1881:  write concat(1bh,'l',80h),' '
1882:  yyy=ci
1883:  end
1884:
1885:
1886:
```

280722-43

```
                                    *WR12*
                                    ******

curhome;cleareos
write concat(1bh,'L',83h),'LOADING....'
write concat(1bh,'L',80h),''
define PROC wr12= DO
define CmAR yyy
define BYTE temp
curhome ;cleareos
write concat(1bh,'L',81h),'                         WRITE REGISTER 12'
write concat(1bh,'L',81h),'                         ******************'
write concat(1bh,'L',80h),'_____________________________________________'
write concat(1bh,'L',80h),'¶ D7 ¶ D6 ¶ D5 ¶ D4 ¶ D3 ¶ D2 ¶ D1 ¶ D0 ¶'
write concat(1bh,'L',80h),'---------------------------------------------'
write '     D7  =   TC7'
write '          D6  =   TC6'
write '               D5  =   TC5'
write '                    D4  =   TC4'
write '                         D3  =   TC3'
write '                              D2  =   TC2'
write '                                   D1  =   TC1'
write '                                        D0  =   TC0'
write ' '
write 'NOTE: Lower Byte of TIME CONSTANT'
write concat(1bh,'L',83h),'   PLEASE TYPE THE VALUE TO BE WRITTEN and <cr>'
write concat(1bh,'L',80h), ' '
temp = gethex
write 'a'
port(wreg) = 12t
port(wreg) = temp
write concat(1bh,'L',83H),'           PLEASE HIT ANY KEY TO RETURN'
write concat(1bh,'l',80h), ' '
yyy=ci
end


                                    *WR13*
                                    ******

curhome;cleareos
write concat(1bh,'L',83h),'LOADING....'
write concat(1bh,'L',80h),''
define PROC wr13= DO
define CmAR yyy
define BYTE temp
curhome ;cleareos
write concat(1bh,'L',81h),'                         WRITE REGISTER 13'
write concat(1bh,'L',81h),'                         ******************'
write concat(1bh,'L',80h),'_____________________________________________'
write concat(1bh,'L',80h),'¶ D7 ¶ D6 ¶ D5 ¶ D4 ¶ D3 ¶ D2 ¶ D1 ¶ D0 ¶'
write concat(1bh,'L',80h),'---------------------------------------------'
write '     D7  =   TC15'
write '          D6  =   TC14'
write '               D5  =   TC13'
write '                    D4  =   TC12'
write '                         D3  =   TC11'
write '                              D2  =   TC10'
write '                                   D1  =   TC9'
write '                                        D0  =   TC8'
write ' '
write 'NOTE: Upper Byte of TIME CONSTANT'
write concat(1bh,'L',83h),'   PLEASE TYPE THE VALUE TO BE WRITTEN and <cr>'
write concat(1bh,'L',80h),' '
temp = gethex
write 'H'
port(wreg) = 13t
port(wreg) = temp
write concat(1BH,'L',83H),'           PLEASE HIT ANY KEY TO RETURN'
write concat(1bh,'l',80h), ' '
yyy=ci
end


                                    *WR14*
                                    ******

curhome;cleareos
write concat(1bh,'L',83h),'LOADING....'
write concat(1bh,'L',80h),''
```

280722-44

```
1969:  define PROC wr14 = DO
1970:  define CHAR yyy
1971:  define BYTE temp
1972:  curhome ;cleareos
1973:  write concat(1bh,'L',81h),'                    WRITE REGISTER 14'
1974:  write concat(1bh,'L',81h),'                    ****************'
1975:  write concat(1bh,'L',80h),'-------------------------------------------------'
1976:  write concat(1bh,'L',80h),'¶ D7  ¶ D6  ¶ D5  ¶ D4  ¶ D3  ¶ D2  ¶ D1  ¶ D0  ¶'
1977:  write concat(1bh,'L',80h),'-------------------------------------------------'
1978:  write '    D7     D6     D5  =  0 0 0  =  NULL COMMAND'
1979:  write '                      =  0 0 1  =  ENTER SEARCH MODE'
1980:  write '                      =  0 1 0  =  RESET MISSSING CLOCK'
1981:  write '                      =  0 1 1  =  DISABLE DPLL'
1982:  write '                      =  1 0 0  =  SET SOURCE = BR GENERATOR'
1983:  write '                      =  1 0 1  =  SET SOURCE = (NOT)RT x C'
1984:  write '                      =  1 1 0  =  SET FM MODE'
1985:  write '                      =  1 1 1  =  SET NRZI MODE'
1986:  write '                      D4  =   LOCAL LOOPBACK'
1987:  write '                            D3  =   AUOT ECHO'
1988:  write '                                  D2  =   (NOT)DTR REQUEST FUNCTION'
1989:  write '                                        D1  =   BR GENERATOR SOURCE'
1990:  write '                                              D0  =   BR GEN. ENABLE'
1991:  write concat(1bh,'L',83h),'   PLEASE TYPE THE VALUE TO BE WRITTEN and <cr>'
1992:  write concat(1bh,'L',80h),' '
1993:  temp = gethex
1994:  write 'n'
1995:  port(wreg) = 14t
1996:  port(wreg) = temp
1997:  write concat(1Bh,'L',83h),'          PLEASE HIT ANY KEY TO RETURN '
1998:  write concat(1bh,'1',80h), ' '
1999:  yyy=ci
2000:  end
2001:  curhome;cleareos
2002:  write concat(1bh,'L',83h),'LOADING....'
2003:  write concat(1bh,'L',80h),''
2004:  define PROC wr14 = DO
2005:  define CHAR yyy
2006:  define BYTE temp
2007:  curhome ;cleareos
2008:  write concat(1bh,'L',81h),'                    WRITE REGISTER 14'
2009:  write concat(1bh,'L',81h),'                    ****************'
2010:  write concat(1bh,'L',80h),'-------------------------------------------------'
2011:  write concat(1bh,'L',80h),'¶ D7  ¶ D6  ¶ D5  ¶ D4  ¶ D3  ¶ D2  ¶ D1  ¶ D0  ¶'
2012:  write concat(1bh,'L',80h),'-------------------------------------------------'
2013:  write '    D7     D6     D5  =  0 0 0  =  NULL COMMAND'
2014:  write '                      =  0 0 1  =  ENTER SEARCH MODE'
2015:  write '                      =  0 1 0  =  RESET MISSSING CLOCK'
2016:  write '                      =  0 1 1  =  DISABLE DPLL'
2017:  write '                      =  1 0 0  =  SET SOURCE = BR GENERATOR'
2018:  write '                      =  1 0 1  =  SET SOURCE = (NOT)RT x C'
2019:  write '                      =  1 1 0  =  SET FM MODE'
2020:  write '                      =  1 1 1  =  SET NRZI MODE'
2021:  write '                      D4  =   LOCAL LOOPBACK'
2022:  write '                            D3  =   AUOT ECHO'
2023:  write '                                  D2  =   (NOT)DTR REQUEST FUNCTION'
2024:  write '                                        D1  =   BR GENERATOR SOURCE'
2025:  write '                                              D0  =   BR GEN. ENABLE'
2026:  write concat(1bh,'L',83h),'   PLEASE TYPE THE VALUE TO BE WRITTEN and <cr>'
2027:  write concat(1bh,'L',80h),' '
2028:  temp = gethex
2029:  write 'n'
2030:  port(wreg) = 14t
2031:  port(wreg) = temp
2032:  write concat(1Bh,'L',83h),'          PLEASE HIT ANY KEY TO RETURN '
2033:  write concat(1bh,'1',80h), ' '
2034:  yyy=ci
2035:  end
2036:
2037:
2038:
2039:                                  *WR15*
2040:                                  ******
2041:
2042:  curhome;cleareos
2043:  write concat(1bh,'L',83h),'LOADING....'
2044:  write concat(1bh,'L',80h),''
2045:  define PROC wr15= DO
2046:  define CHAR yyy
2047:  define BYTE temp
2048:  curhome;cleareos
```

280722-45

```
2049:   write concat(1bh,'L',81h),'                    WRITE REGISTER 15'
2050:   write concat(1bh,'L',81h),'                    ****************'
2051:   write concat(1bh,'L',80h),'_____________________________________________________'

2052:   write concat(1bh,'L',80h),'| D7  | D6  | D5  | D4  | D3  | D2  | D1  | D0  |

2053:   write concat(1bh,'L',80h),'-----------------------------------------------------

2054:   write '    D7  =   BREAK/ABORT IE'
2055:   write '          D6  =   Tx UNDERRUN/EOM IE'
2056:   write '                D5  =   CTS IE'
2057:   write '                      D4  =   SYNC/HUNT IE'
2058:   write '                            D3  =   CD IE'
2059:   write '                                  D2  =   0'
2060:   write '                                        D1  =   ZERO COUNT IE'
2061:   write '                                              D0  =   0'
2062:   write concat(1bh,'L',83h),'   PLEASE TYPE THE VALUE TO BE WRITTEN and <cr>'
2063:   write concat(1bh,'L',80h),' '
2064:   temp = gethex
2065:   write 'H'
2066:   port(wreg) = 15t
2067:   port(wreg) = temp
2068:   write concat(1BH,'L',83H),'          PLEASE HIT ANY KEY TO RETURN'
2069:   write concat(1bh,'1',80h),' '
2070:   yyy=ci
2071:   end
```

280722-46